Microsoft

Microsoft Official Course

PL-300T00

Microsoft Power BI Data Analyst

PL-300T00
Microsoft Power BI Data Analyst

1 http://www.microsoft.com/trademarks

MICROSOFT LICENSE TERMS

MICROSOFT INSTRUCTOR-LED COURSEWARE

These license terms are an agreement between Microsoft Corporation (or based on where you live, one of its affiliates) and you. Please read them. They apply to your use of the content accompanying this agreement which includes the media on which you received it, if any. These license terms also apply to Trainer Content and any updates and supplements for the Licensed Content unless other terms accompany those items. If so, those terms apply.

BY ACCESSING, DOWNLOADING OR USING THE LICENSED CONTENT, YOU ACCEPT THESE TERMS. IF YOU DO NOT ACCEPT THEM, DO NOT ACCESS, DOWNLOAD OR USE THE LICENSED CONTENT.

If you comply with these license terms, you have the rights below for each license you acquire.

1. **DEFINITIONS.**
    1. "Authorized Learning Center" means a Microsoft Imagine Academy (MSIA) Program Member, Microsoft Learning Competency Member, or such other entity as Microsoft may designate from time to time.
    2. "Authorized Training Session" means the instructor-led training class using Microsoft Instructor-Led Courseware conducted by a Trainer at or through an Authorized Learning Center.
    3. "Classroom Device" means one (1) dedicated, secure computer that an Authorized Learning Center owns or controls that is located at an Authorized Learning Center's training facilities that meets or exceeds the hardware level specified for the particular Microsoft Instructor-Led Courseware.
    4. "End User" means an individual who is (i) duly enrolled in and attending an Authorized Training Session or Private Training Session, (ii) an employee of an MPN Member (defined below), or (iii) a Microsoft full-time employee, a Microsoft Imagine Academy (MSIA) Program Member, or a Microsoft Learn for Educators – Validated Educator.
    5. "Licensed Content" means the content accompanying this agreement which may include the Microsoft Instructor-Led Courseware or Trainer Content.
    6. "Microsoft Certified Trainer" or "MCT" means an individual who is (i) engaged to teach a training session to End Users on behalf of an Authorized Learning Center or MPN Member, and (ii) currently certified as a Microsoft Certified Trainer under the Microsoft Certification Program.
    7. "Microsoft Instructor-Led Courseware" means the Microsoft-branded instructor-led training course that educates IT professionals, developers, students at an academic institution, and other learners on Microsoft technologies. A Microsoft Instructor-Led Courseware title may be branded as MOC, Microsoft Dynamics, or Microsoft Business Group courseware.
    8. "Microsoft Imagine Academy (MSIA) Program Member" means an active member of the Microsoft Imagine Academy Program.
    9. "Microsoft Learn for Educators – Validated Educator" means an educator who has been validated through the Microsoft Learn for Educators program as an active educator at a college, university, community college, polytechnic or K-12 institution.
    10. "Microsoft Learning Competency Member" means an active member of the Microsoft Partner Network program in good standing that currently holds the Learning Competency status.
    11. "MOC" means the "Official Microsoft Learning Product" instructor-led courseware known as Microsoft Official Course that educates IT professionals, developers, students at an academic institution, and other learners on Microsoft technologies.
    12. "MPN Member" means an active Microsoft Partner Network program member in good standing.

13. "Personal Device" means one (1) personal computer, device, workstation or other digital electronic device that you personally own or control that meets or exceeds the hardware level specified for the particular Microsoft Instructor-Led Courseware.
14. "Private Training Session" means the instructor-led training classes provided by MPN Members for corporate customers to teach a predefined learning objective using Microsoft Instructor-Led Courseware. These classes are not advertised or promoted to the general public and class attendance is restricted to individuals employed by or contracted by the corporate customer.
15. "Trainer" means (i) an academically accredited educator engaged by a Microsoft Imagine Academy Program Member to teach an Authorized Training Session, (ii) an academically accredited educator validated as a Microsoft Learn for Educators – Validated Educator, and/or (iii) a MCT.
16. "Trainer Content" means the trainer version of the Microsoft Instructor-Led Courseware and additional supplemental content designated solely for Trainers' use to teach a training session using the Microsoft Instructor-Led Courseware. Trainer Content may include Microsoft PowerPoint presentations, trainer preparation guide, train the trainer materials, Microsoft One Note packs, classroom setup guide and Pre-release course feedback form. To clarify, Trainer Content does not include any software, virtual hard disks or virtual machines.

2. **USE RIGHTS.** The Licensed Content is licensed, not sold. The Licensed Content is licensed on a **one copy per user basis**, such that you must acquire a license for each individual that accesses or uses the Licensed Content.

- 2.1 Below are five separate sets of use rights. Only one set of rights apply to you.

  1. **If you are a Microsoft Imagine Academy (MSIA) Program Member:**
     1. Each license acquired on behalf of yourself may only be used to review one (1) copy of the Microsoft Instructor-Led Courseware in the form provided to you. If the Microsoft Instructor-Led Courseware is in digital format, you may install one (1) copy on up to three (3) Personal Devices. You may not install the Microsoft Instructor-Led Courseware on a device you do not own or control.
     2. For each license you acquire on behalf of an End User or Trainer, you may either:
        1. distribute one (1) hard copy version of the Microsoft Instructor-Led Courseware to one (1) End User who is enrolled in the Authorized Training Session, and only immediately prior to the commencement of the Authorized Training Session that is the subject matter of the Microsoft Instructor-Led Courseware being provided, **or**
        2. provide one (1) End User with the unique redemption code and instructions on how they can access one (1) digital version of the Microsoft Instructor-Led Courseware, **or**
        3. provide one (1) Trainer with the unique redemption code and instructions on how they can access one (1) Trainer Content.
     3. For each license you acquire, you must comply with the following:
        1. you will only provide access to the Licensed Content to those individuals who have acquired a valid license to the Licensed Content,
        2. you will ensure each End User attending an Authorized Training Session has their own valid licensed copy of the Microsoft Instructor-Led Courseware that is the subject of the Authorized Training Session,
        3. you will ensure that each End User provided with the hard-copy version of the Microsoft Instructor-Led Courseware will be presented with a copy of this agreement and each End

User will agree that their use of the Microsoft Instructor-Led Courseware will be subject to the terms in this agreement prior to providing them with the Microsoft Instructor-Led Courseware. Each individual will be required to denote their acceptance of this agreement in a manner that is enforceable under local law prior to their accessing the Microsoft Instructor-Led Courseware,

4. you will ensure that each Trainer teaching an Authorized Training Session has their own valid licensed copy of the Trainer Content that is the subject of the Authorized Training Session,
5. you will only use qualified Trainers who have in-depth knowledge of and experience with the Microsoft technology that is the subject of the Microsoft Instructor-Led Courseware being taught for all your Authorized Training Sessions,
6. you will only deliver a maximum of 15 hours of training per week for each Authorized Training Session that uses a MOC title, and
7. you acknowledge that Trainers that are not MCTs will not have access to all of the trainer resources for the Microsoft Instructor-Led Courseware.

2. **If you are a Microsoft Learning Competency Member:**
   1. Each license acquire may only be used to review one (1) copy of the Microsoft Instructor-Led Courseware in the form provided to you. If the Microsoft Instructor-Led Courseware is in digital format, you may install one (1) copy on up to three (3) Personal Devices. You may not install the Microsoft Instructor-Led Courseware on a device you do not own or control.
   2. For each license you acquire on behalf of an End User or MCT, you may either:
      1. distribute one (1) hard copy version of the Microsoft Instructor-Led Courseware to one (1) End User attending the Authorized Training Session and only immediately prior to the commencement of the Authorized Training Session that is the subject matter of the Microsoft Instructor-Led Courseware provided, **or**
      2. provide one (1) End User attending the Authorized Training Session with the unique redemption code and instructions on how they can access one (1) digital version of the Microsoft Instructor-Led Courseware, **or**
      3. you will provide one (1) MCT with the unique redemption code and instructions on how they can access one (1) Trainer Content.
   3. For each license you acquire, you must comply with the following:
      1. you will only provide access to the Licensed Content to those individuals who have acquired a valid license to the Licensed Content,
      2. you will ensure that each End User attending an Authorized Training Session has their own valid licensed copy of the Microsoft Instructor-Led Courseware that is the subject of the Authorized Training Session,
      3. you will ensure that each End User provided with a hard-copy version of the Microsoft Instructor-Led Courseware will be presented with a copy of this agreement and each End User will agree that their use of the Microsoft Instructor-Led Courseware will be subject to the terms in this agreement prior to providing them with the Microsoft Instructor-Led Courseware. Each individual will be required to denote their acceptance of this agreement in a manner that is enforceable under local law prior to their accessing the Microsoft Instructor-Led Courseware,

4. you will ensure that each MCT teaching an Authorized Training Session has their own valid licensed copy of the Trainer Content that is the subject of the Authorized Training Session,
5. you will only use qualified MCTs who also hold the applicable Microsoft Certification credential that is the subject of the MOC title being taught for all your Authorized Training Sessions using MOC,
6. you will only provide access to the Microsoft Instructor-Led Courseware to End Users, and
7. you will only provide access to the Trainer Content to MCTs.

3. **If you are a MPN Member:**
   1. Each license acquired on behalf of yourself may only be used to review one (1) copy of the Microsoft Instructor-Led Courseware in the form provided to you. If the Microsoft Instructor-Led Courseware is in digital format, you may install one (1) copy on up to three (3) Personal Devices. You may not install the Microsoft Instructor-Led Courseware on a device you do not own or control.
   2. For each license you acquire on behalf of an End User or Trainer, you may either:
      1. distribute one (1) hard copy version of the Microsoft Instructor-Led Courseware to one (1) End User attending the Private Training Session, and only immediately prior to the commencement of the Private Training Session that is the subject matter of the Microsoft Instructor-Led Courseware being provided, **or**
      2. provide one (1) End User who is attending the Private Training Session with the unique redemption code and instructions on how they can access one (1) digital version of the Microsoft Instructor-Led Courseware, **or**
      3. you will provide one (1) Trainer who is teaching the Private Training Session with the unique redemption code and instructions on how they can access one (1) Trainer Content.
   3. For each license you acquire, you must comply with the following:
      1. you will only provide access to the Licensed Content to those individuals who have acquired a valid license to the Licensed Content,
      2. you will ensure that each End User attending an Private Training Session has their own valid licensed copy of the Microsoft Instructor-Led Courseware that is the subject of the Private Training Session,
      3. you will ensure that each End User provided with a hard copy version of the Microsoft Instructor-Led Courseware will be presented with a copy of this agreement and each End User will agree that their use of the Microsoft Instructor-Led Courseware will be subject to the terms in this agreement prior to providing them with the Microsoft Instructor-Led Courseware. Each individual will be required to denote their acceptance of this agreement in a manner that is enforceable under local law prior to their accessing the Microsoft Instructor-Led Courseware,
      4. you will ensure that each Trainer teaching an Private Training Session has their own valid licensed copy of the Trainer Content that is the subject of the Private Training Session,

5. you will only use qualified Trainers who hold the applicable Microsoft Certification credential that is the subject of the Microsoft Instructor-Led Courseware being taught for all your Private Training Sessions,

6. you will only use qualified MCTs who hold the applicable Microsoft Certification credential that is the subject of the MOC title being taught for all your Private Training Sessions using MOC,

7. you will only provide access to the Microsoft Instructor-Led Courseware to End Users, and

8. you will only provide access to the Trainer Content to Trainers.

4. **If you are an End User:**
For each license you acquire, you may use the Microsoft Instructor-Led Courseware solely for your personal training use. If the Microsoft Instructor-Led Courseware is in digital format, you may access the Microsoft Instructor-Led Courseware online using the unique redemption code provided to you by the training provider and install and use one (1) copy of the Microsoft Instructor-Led Courseware on up to three (3) Personal Devices. You may also print one (1) copy of the Microsoft Instructor-Led Courseware. You may not install the Microsoft Instructor-Led Courseware on a device you do not own or control.

5. **If you are a Trainer.**

1. For each license you acquire, you may install and use one (1) copy of the Trainer Content in the form provided to you on one (1) Personal Device solely to prepare and deliver an Authorized Training Session or Private Training Session, and install one (1) additional copy on another Personal Device as a backup copy, which may be used only to reinstall the Trainer Content. You may not install or use a copy of the Trainer Content on a device you do not own or control. You may also print one (1) copy of the Trainer Content solely to prepare for and deliver an Authorized Training Session or Private Training Session.

2. If you are an MCT, you may customize the written portions of the Trainer Content that are logically associated with instruction of a training session in accordance with the most recent version of the MCT agreement.

3. If you elect to exercise the foregoing rights, you agree to comply with the following: (i) customizations may only be used for teaching Authorized Training Sessions and Private Training Sessions, and (ii) all customizations will comply with this agreement. For clarity, any use of "customize" refers only to changing the order of slides and content, and/or not using all the slides or content, it does not mean changing or modifying any slide or content.

- 2.2 **Separation of Components.** The Licensed Content is licensed as a single unit and you may not separate their components and install them on different devices.
- 2.3 **Redistribution of Licensed Content.** Except as expressly provided in the use rights above, you may not distribute any Licensed Content or any portion thereof (including any permitted modifications) to any third parties without the express written permission of Microsoft.
- 2.4 **Third Party Notices.** The Licensed Content may include third party code that Microsoft, not the third party, licenses to you under this agreement. Notices, if any, for the third party code are included for your information only.
- 2.5 **Additional Terms.** Some Licensed Content may contain components with additional terms, conditions, and licenses regarding its use. Any non-conflicting terms in those conditions and licenses also apply to your use of that respective component and supplements the terms described in this agreement.

3. **LICENSED CONTENT BASED ON PRE-RELEASE TECHNOLOGY.** If the Licensed Content's subject matter is based on a pre-release version of Microsoft technology ("**Pre-release**"), then in addition to the other provisions in this agreement, these terms also apply:

    1. **Pre-Release Licensed Content.** This Licensed Content subject matter is on the Pre-release version of the Microsoft technology. The technology may not work the way a final version of the technology will and we may change the technology for the final version. We also may not release a final version. Licensed Content based on the final version of the technology may not contain the same information as the Licensed Content based on the Pre-release version. Microsoft is under no obligation to provide you with any further content, including any Licensed Content based on the final version of the technology.

    2. **Feedback.** If you agree to give feedback about the Licensed Content to Microsoft, either directly or through its third party designee, you give to Microsoft without charge, the right to use, share and commercialize your feedback in any way and for any purpose. You also give to third parties, without charge, any patent rights needed for their products, technologies and services to use or interface with any specific parts of a Microsoft technology, Microsoft product, or service that includes the feedback. You will not give feedback that is subject to a license that requires Microsoft to license its technology, technologies, or products to third parties because we include your feedback in them. These rights survive this agreement.

    3. **Pre-release Term.** If you are an Microsoft Imagine Academy Program Member, Microsoft Learning Competency Member, MPN Member, Microsoft Learn for Educators – Validated Educator, or Trainer, you will cease using all copies of the Licensed Content on the Pre-release technology upon (i) the date which Microsoft informs you is the end date for using the Licensed Content on the Pre-release technology, or (ii) sixty (60) days after the commercial release of the technology that is the subject of the Licensed Content, whichever is earliest ("**Pre-release term**"). Upon expiration or termination of the Pre-release term, you will irretrievably delete and destroy all copies of the Licensed Content in your possession or under your control.

4. **SCOPE OF LICENSE.** The Licensed Content is licensed, not sold. This agreement only gives you some rights to use the Licensed Content. Microsoft reserves all other rights. Unless applicable law gives you more rights despite this limitation, you may use the Licensed Content only as expressly permitted in this agreement. In doing so, you must comply with any technical limitations in the Licensed Content that only allows you to use it in certain ways. Except as expressly permitted in this agreement, you may not:

    - access or allow any individual to access the Licensed Content if they have not acquired a valid license for the Licensed Content,
    - alter, remove or obscure any copyright or other protective notices (including watermarks), branding or identifications contained in the Licensed Content,
    - modify or create a derivative work of any Licensed Content,
    - publicly display, or make the Licensed Content available for others to access or use,
    - copy, print, install, sell, publish, transmit, lend, adapt, reuse, link to or post, make available or distribute the Licensed Content to any third party,
    - work around any technical limitations in the Licensed Content, or
    - reverse engineer, decompile, remove or otherwise thwart any protections or disassemble the Licensed Content except and only to the extent that applicable law expressly permits, despite this limitation.

5. **RESERVATION OF RIGHTS AND OWNERSHIP.** Microsoft reserves all rights not expressly granted to you in this agreement. The Licensed Content is protected by copyright and other intellectual property

laws and treaties. Microsoft or its suppliers own the title, copyright, and other intellectual property rights in the Licensed Content.

6. **EXPORT RESTRICTIONS.** The Licensed Content is subject to United States export laws and regulations. You must comply with all domestic and international export laws and regulations that apply to the Licensed Content. These laws include restrictions on destinations, end users and end use. For additional information, see www.microsoft.com/exporting.
7. **SUPPORT SERVICES.** Because the Licensed Content is provided "as is", we are not obligated to provide support services for it.
8. **TERMINATION.** Without prejudice to any other rights, Microsoft may terminate this agreement if you fail to comply with the terms and conditions of this agreement. Upon termination of this agreement for any reason, you will immediately stop all use of and delete and destroy all copies of the Licensed Content in your possession or under your control.
9. **LINKS TO THIRD PARTY SITES.** You may link to third party sites through the use of the Licensed Content. The third party sites are not under the control of Microsoft, and Microsoft is not responsible for the contents of any third party sites, any links contained in third party sites, or any changes or updates to third party sites. Microsoft is not responsible for webcasting or any other form of transmission received from any third party sites. Microsoft is providing these links to third party sites to you only as a convenience, and the inclusion of any link does not imply an endorsement by Microsoft of the third party site.
10. **ENTIRE AGREEMENT.** This agreement, and any additional terms for the Trainer Content, updates and supplements are the entire agreement for the Licensed Content, updates and supplements.
11. **APPLICABLE LAW.**
    1. United States. If you acquired the Licensed Content in the United States, Washington state law governs the interpretation of this agreement and applies to claims for breach of it, regardless of conflict of laws principles. The laws of the state where you live govern all other claims, including claims under state consumer protection laws, unfair competition laws, and in tort.
    2. Outside the United States. If you acquired the Licensed Content in any other country, the laws of that country apply.
12. **LEGAL EFFECT.** This agreement describes certain legal rights. You may have other rights under the laws of your country. You may also have rights with respect to the party from whom you acquired the Licensed Content. This agreement does not change your rights under the laws of your country if the laws of your country do not permit it to do so.
13. **DISCLAIMER OF WARRANTY. THE LICENSED CONTENT IS LICENSED "AS-IS" AND "AS AVAILABLE." YOU BEAR THE RISK OF USING IT. MICROSOFT AND ITS RESPECTIVE AFFILIATES GIVES NO EXPRESS WARRANTIES, GUARANTEES, OR CONDITIONS. YOU MAY HAVE ADDITIONAL CONSUMER RIGHTS UNDER YOUR LOCAL LAWS WHICH THIS AGREEMENT CANNOT CHANGE. TO THE EXTENT PERMITTED UNDER YOUR LOCAL LAWS, MICROSOFT AND ITS RESPECTIVE AFFILIATES EXCLUDES ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT.**
14. **LIMITATION ON AND EXCLUSION OF REMEDIES AND DAMAGES. YOU CAN RECOVER FROM MICROSOFT, ITS RESPECTIVE AFFILIATES AND ITS SUPPLIERS ONLY DIRECT DAMAGES UP TO US$5.00. YOU CANNOT RECOVER ANY OTHER DAMAGES, INCLUDING CONSEQUENTIAL, LOST PROFITS, SPECIAL, INDIRECT OR INCIDENTAL DAMAGES.**

This limitation applies to

- anything related to the Licensed Content, services, content (including code) on third party Internet sites or third-party programs; and
- claims for breach of contract, breach of warranty, guarantee or condition, strict liability, negligence, or other tort to the extent permitted by applicable law.

It also applies even if Microsoft knew or should have known about the possibility of the damages. The above limitation or exclusion may not apply to you because your country may not allow the exclusion or limitation of incidental, consequential, or other damages.

Please note: As this Licensed Content is distributed in Quebec, Canada, some of the clauses in this agreement are provided below in French.

Remarque : Ce le contenu sous licence étant distribué au Québec, Canada, certaines des clauses dans ce contrat sont fournies ci-dessous en français.

EXONÉRATION DE GARANTIE. Le contenu sous licence visé par une licence est offert « tel quel ». Toute utilisation de ce contenu sous licence est à votre seule risque et péril. Microsoft n'accorde aucune autre garantie expresse. Vous pouvez bénéficier de droits additionnels en vertu du droit local sur la protection dues consommateurs, que ce contrat ne peut modifier. La ou elles sont permises par le droit locale, les garanties implicites de qualité marchande, d'adéquation à un usage particulier et d'absence de contre-façon sont exclues.

LIMITATION DES DOMMAGES-INTÉRÊTS ET EXCLUSION DE RESPONSABILITÉ POUR LES DOMMAGES. Vous pouvez obtenir de Microsoft et de ses fournisseurs une indemnisation en cas de dommages directs uniquement à hauteur de 5,00 $ US. Vous ne pouvez prétendre à aucune indemnisation pour les autres dommages, y compris les dommages spéciaux, indirects ou accessoires et pertes de bénéfices.

Cette limitation concerne:

- tout ce qui est relié au le contenu sous licence, aux services ou au contenu (y compris le code) figurant sur des sites Internet tiers ou dans des programmes tiers; et.
- les réclamations au titre de violation de contrat ou de garantie, ou au titre de responsabilité stricte, de négligence ou d'une autre faute dans la limite autorisée par la loi en vigueur.

Elle s'applique également, même si Microsoft connaissait ou devrait connaître l'éventualité d'un tel dommage. Si votre pays n'autorise pas l'exclusion ou la limitation de responsabilité pour les dommages indirects, accessoires ou de quelque nature que ce soit, il se peut que la limitation ou l'exclusion ci-dessus ne s'appliquera pas à votre égard.

EFFET JURIDIQUE. Le présent contrat décrit certains droits juridiques. Vous pourriez avoir d'autres droits prévus par les lois de votre pays. Le présent contrat ne modifie pas les droits que vous confèrent les lois de votre pays si celles-ci ne le permettent pas.

Revised April 2019

Contents

Module 0 Introduction

Welcome to the Course

Introduction

This course has been created by the Microsoft World Wide Learning team to support exam PL-300: Microsoft Power BI Data Analyst.

Learning Objectives

In this module you will learn:

- About this course
- About the audience
- Course pre-requisites
- Understand the PL-300 certification

Course Introduction

This module provides an overview of the following:

- About this course
- Course agenda
- Course audience
- Course pre-requisites
- PL-300: Microsoft Power BI Data Analyst

About this course

In this course, you will learn about, and apply, the various methods and best practices that are in line with business and technical requirements for ingesting, modeling, visualizing, and analyzing data.

You will also learn about the management aspects of Power BI, including workspaces and datasets, and then learn how to share, distribute, and appropriately secure Power BI assets.

Course Agenda

At the end of this course, the student will learn:

Module 1: Get Started with Microsoft Data Analytics

This module explores the different roles in the data space, outlines the important roles and responsibilities of a Data Analysts, and then explores the landscape of the Power BI portfolio.

Module Objectives:

At the end of this module, the students will be able to:

- Identify the different roles in the data space
- Identify the tasks that are performed by a Data Analyst
- Describe the Power BI landscape of products and services
- Tour the Power BI Service

Module 2: Prepare Data in Power BI

This module explores identifying and retrieving data from various data sources. You will also learn the options for connectivity and data storage, and understand the difference and performance implications of importing or connecting directly to data.

Module Objectives:

At the end of this module, the students will be able to:

- Identify and retrieve data from different data sources
- Understand the different connection methods and their performance implications
- Optimize query performance
- Resolve data import errors

Module 3: Clean, Transform, and Load Data in Power BI

This module teaches you the process of profiling and understanding the condition of the data. They will learn how to identify anomalies, look at the size and shape of their data, and perform the proper data cleaning and transforming steps to prepare the data for loading into the model.

Module Objectives:

At the end of this module, the students will be able to:

- Apply data shape transformations
- Enhance the data structure
- Profile and examine the data

Module 4: Design a Data Model in Power BI

This module teaches you the fundamental concepts of designing and developing a data model for proper performance and scalability. This module will also help you understand and tackle many of the common data modeling issues, including relationships, security, and performance.

Module Objectives:

At the end of this module, the students will be able to:

- Understand the basics of data modeling
- Define relationships and their cardinality
- Implement Dimensions and hierarchies
- Create histograms and rankings

Module 5: Creating Model Calculations using DAX in Power BI

This module introduces you to the world of DAX and its true power for enhancing a model. You will learn about aggregations and the concepts of Measures, calculated columns and tables, and Time Intelligence functions to solve calculation and data analysis problems.

Module Objectives:

At the end of this module, the students will be able to:

- Understand DAX
- Use DAX for simple formulas and expressions.
- Create calculated tables and columns
- Build simple measures
- Work with Time Intelligence and Key Performance Indicators

Module 6: Optimizing Model Performance

In this module you are introduced to the steps, processes, concepts, and data modeling best practices necessary to optimize a data model for enterprise-level performance.

Module Objectives:

At the end of this module, the students will be able to:

- Understand the importance of variables
- Enhance the data model
- Optimize storage

Module 7: Creating Reports

This module introduces you to the fundamental concepts and principles of designing and building a report, including selecting the correct visuals, designing a page layout, and applying basic but critical functionality This important topic of designing for accessibility is also covered.

Module Objectives:

At the end of this module, the students will be able to:

- Design a page layout
- Select and add appropriate visualization type
- Add basic report functionality
- Add basic report navigation and interactions
- Improve report performance

Module 8: Creating Dashboards

In this module you will learn about dashboards, and how to tell a compelling story through the use of dashboards. You will be introduced to features and functionality and how to enhance dashboards for usability and insights.

Module Objectives:

At the end of this module, the students will be able to:

- Create a dashboard
- Understand real-time dashboards
- Enhance the dashboard usability

Module 9: Identify Patterns and Trends

This module helps you apply additional features to enhance the report for analytical insights in the data, equipping you with the steps to use the report for actual data analysis. You will also perform advanced analytics using AI visuals on the report for even deeper and meaningful data insights.

Module Objectives:

At the end of this module, the students will be able to:

- Explore statistical summary
- Use the Analyze feature
- Identify outliers in the data
- Use the AI visuals
- Use the Advanced Analytics custom visual

Module 10: Create and Manage Workspaces

This module will introduce you to Workspaces, including how to create and them. You will also learn how to share content, including reports and dashboards, and then learn how to distribute an App.

Module Objectives:

At the end of this module, the students will be able to:

- Create and manage a workspace
- Understand workspace collaboration
- Monitor usage and performance
- Distribute an App

Module 11: Manage Files and Datasets in Power BI

In this module you will learn the concepts of managing Power BI assets, including datasets and workspaces. You will also publish datasets to the Power BI service, then refresh and secure them.

Module Objectives:

At the end of this module, the students will be able to:

- Configure dataset refresh
- Create and work with parameters
- Manage datasets
- Troubleshoot gateway connectivity

Module 12: Row-level Security

This module teaches you the steps for implementing and configuring security in Power BI to secure Power BI assets.

Module Objectives:

At the end of this module, the students will be able to:

- Understand aspects of Power BI security
- Configure static and dynamic row-level

Course Audience

Primary Audience

The audience for this course are data professionals and business intelligence professionals who want to learn how to accurately perform data analysis using Power BI.

Secondary Audience

The secondary audience for this course are individuals who develop reports that visualize data from the data platform technologies that exist on both in the cloud and on-premises.

Course Prerequisites

In addition to their professional experience, students who take this training should have technical knowledge equivalent to the following courses:

- Azure Fundamentals

Microsoft Certification

Data Analyst Associate

Data Analysts enable businesses to maximize the value of their data assets using Power BI by providing meaningful business value and analysis through easy-to-understand visualizations. To gain this certification, you must bass the following exam:

- PL-300: Microsoft Power BI Data Analyst

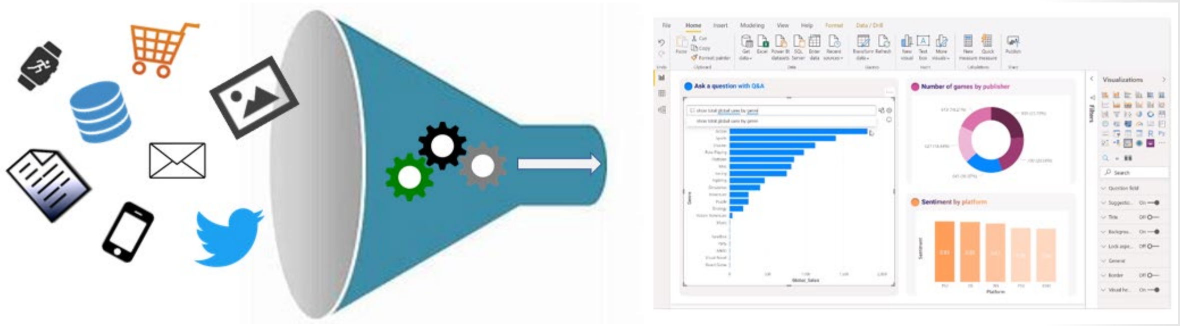
This course is used to prepare for exam PL-300.

Module 1 Getting Started with Microsoft Data Analytics

Data analytics and Microsoft

Introduction

As a data analyst, you are on a journey. Think about all the data that is being generated each day and that is available in an organization, from transactional data in a traditional database, telemetry data from services you use, to signals you get from different areas like social media.



For example, retail businesses of today collect and store massive amounts of data, which track the items you browsed and purchased, the pages you've visited on their site, the aisles you purchase products from, your spending habits, and much more.

With data and information as the most strategic asset of a business, the underlying challenge that organizations have today is understanding and using their data to positively affect change within the business. Yet, businesses continue to struggle to use their data in a meaningful and productive way, which impacts their ability to act.

A retail business should be able to use their vast amounts of data and information in such a way that impacts the business, including tracking inventory, identifying purchase habits, detecting end-user trends and patterns, recommending purchases, determining price optimizations, and identifying and stopping fraud. You may also be searching for daily/monthly sale patterns. Day over day, week over week and month over month are also common segments to invstigate. How have sales compared to where we were in the same week last year?

They key to unlocking all this data is being able to tell a story with it. In today's highly competitive and fast-paced business world, crafting reports that tell that story is what helps business leaders to really take action on the data. Business decision makers depend on an accurate story to drive better business decisions. The faster a business can make precise decisions, the more competitive they will be and the better advantage they will have. Without the story, it is hard to understand what the data is trying to tell you.

Having data alone is not enough. You need to be able to act on the data to effect change within the business. Whether that is reallocating resources within the business to accommodate a need or being able to identify a failing campaign and knowing when to change course. This is where telling a story with your data is important.

The underlying challenge that businesses face today is understanding and using their data in such a way that impacts their business and ultimately their bottom line. You need to be able to interpret data, clearly understand the meaning behind the metrics, and facilitate trusted business decisions.

This sounds daunting but know that you are not alone. You will want to partner with the data professionals within your organization, such as data engineers and data scientists, to help get the data you need to tell that story. Have them take part in that data journey with you.

Your journey of telling a story with data also ties into building that data culture within your organization. While telling the story is important, *where* that story is told is also crucial, making sure the story is told to the right people. Also, make sure people can discover the story, they know where to find it, and that it is part of the regular interactions.

Data analysis exists to help overcome these challenges and pain points, assisting businesses to find insights and uncover hidden value in troves of data through story telling. As you read on, you will learn how you will utilize and apply analytical skills to go beyond a single report and help to impact and influence your organization by telling stories with data and driving that data culture.

Overview of data analysis

Before data can be used to tell a story, it must first be run through a process that makes it usable in the story. Thus, data analysis is the process of identifying, cleaning, transforming, and modeling data to discover meaningful and useful information. The data is then crafted into a story via reports for analysis to support the critical decision-making process.

As the world becomes more data-driven, storytelling through data analysis is becoming a more vital component and aspect of businesses large and small and is the reason organizations continue to hire data analysts.

Data-driven businesses make decisions based on the story their data tells, and in today's data-driven world, data is not being used to its full potential, a challenge that most business face today. Data analysis is and should be a critical aspect of all organizations to determine the impact to their business, including customer sentiment, market and product research, identify trends, or any other data insights.

While the process of data analysis focuses on the tasks of cleaning, modeling, and visualizing data, the concept of data analysis and how it is important to business should not be understated. To analyze data, core components of analytics are divided in the following categories:

Descriptive analytics

Descriptive analytics helps answer questions about what has happened, based on historical data. Descriptive analytics techniques summarize large datasets to describe outcomes to stakeholders.

By developing KPIs (Key Performance Indicators), these strategies can help track the success or failure of key objectives. Metrics such as return on investment (ROI) are used in many industries. Specialized metrics are developed to track performance in specific industries.

Examples of descriptive analytics include generating reports to provide a view of an organization's sales and financial data.

Diagnostic analytics

Diagnostic analytics helps answer questions about why things happened. Diagnostic analytics techniques supplement more basic descriptive analytics. They take the findings from descriptive analytics and dig deeper to find the cause. The performance indicators are further investigated to discover why they got better or worse. This generally occurs in three steps:

1. Identify anomalies in the data. These may be unexpected changes in a metric or a particular market.

2. Collect data that's related to these anomalies.
3. Use statistical techniques to discover relationships and trends that explain these anomalies.

Predictive analytics

Predictive analytics helps answer questions about what will happen in the future. Predictive analytics techniques use historical data to identify trends and determine if they're likely to recur. Predictive analytical tools provide valuable insight into what may happen in the future. Techniques include a variety of statistical and machine learning techniques such as neural networks, decision trees, and regression.

Prescriptive analytics

Prescriptive analytics helps answer questions about what actions should be taken to achieve a goal or target. By using insights from predictive analytics, data-driven decisions can be made. This technique allows businesses to make informed decisions in the face of uncertainty. Prescriptive analytics techniques rely on machine learning strategies to find patterns in large datasets. By analyzing past decisions and events, the likelihood of different outcomes can be estimated.

Cognitive analytics

Cognitive analytics attempts to draw inferences from existing data and patterns, derive conclusions based on existing knowledge bases, and then add these findings back into the knowledge base for future inferences, a self-learning feedback loop. Cognitive analytics helps you to learn what might happen if circumstances change, and how you might handle these situations.

Inferences aren't structured queries based on a rules database, rather they're unstructured hypotheses gathered from several sources and expressed with varying degrees of confidence. Effective cognitive analytics depends on machine learning algorithms. It uses several NLP (Natural Language Processing) concepts to make sense of previously untapped data sources, such as call center conversation logs and product reviews.

Example

By enabling reporting and data visualizations, a retail business uses descriptive analytics to look at patterns of purchases from previous years to determine what products might be popular next year. They might also look at supporting data to understand why a particular product was popular and if that trend is continuing, determine whether they need to continue to stock that product.

A business might determine that a certain product was popular over a certain timeframe and use analysis to determine whether any marketing efforts or online social activities contributed to the sales increase.

An underlying facet of data analysis is that businesses need to be able to trust its data, and as a practice, the data analysis process take data from trusted sources and shapes it into something that is consumable, meaningful and easily understood from which business decisions can be made. Data analysis enables businesses to fully understand their data through data-driven processes and decisions, allowing businesses to be confident in their decisions.

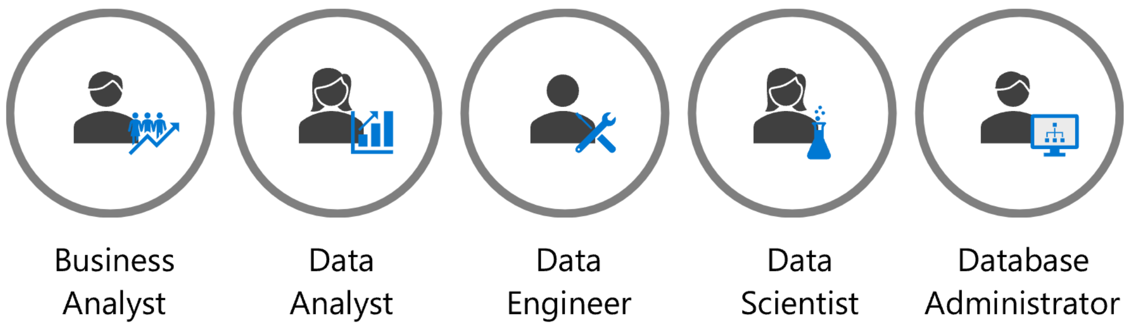
As the amount of data grows, so does the need for data analysts. A data analyst knows how to organize information and distill it into something meaningful and understandable. A data analyst knows how to gather the right data and what to do with it, i.e., the mission of making sense of the data in your data swamp.

Roles in data

As mentioned before, you are not alone. This is a journey and it usually doesn't start with you. The data must come from somewhere. And getting that data into a spot that is usable by you takes some effort and more than likely out of your scope. Especially when we think of the enterprise.

Today's applications and projects can be large and intricate, often utilizing the skills and knowledge of numerous individuals. Each person brings a unique talent and expertise to the table, working together and coordinating tasks and responsibilities in order to see a project through from concept to production.

It wasn't too long ago when roles such as business analysts and business intelligence developers were the de facto norm for data processing and understanding. But the growth and explosion of both the size of data and the different types of data has caused these roles to evolve into more specialized set of skills that modernize and streamline the processes of data engineering and analysis.



The following highlights the different roles in data and the specific responsibility in the overall, end-to-end spectrum of data discovery and understanding.

- Business analyst
- Data analyst
- Data engineer
- Data scientist
- Database administrator

Business Analyst

While there are some similarities between a data analyst and business analyst, the key differentiator between the two roles is what they do with data. A business analyst is closer to the business itself and is a specialist on interpreting the data that comes from the visualization. Often the data analyst and business analyst could be the accountability of a single person.

Data Analyst

A data analyst enables businesses to maximize the value of their data assets through visualization and reporting tools such as Power BI. Data analysts are responsible for profiling, cleaning, and transforming data, designing, and building scalable and performant data models, and enabling and implementing the advanced analytics capabilities into reports for analysis. They work with the appropriate stakeholders to identify appropriate and necessary data and reporting requirements and then are tasked with turning raw data into relevant and meaningful insights.

A data analyst is also responsible for the management of Power BI assets, including reports, dashboards, workspaces, and the underlying datasets used in the reports. They are tasked with implementing and configuring proper security procedures, in conjunction with stakeholder requirements, to ensure the safekeeping of all Power BI assets and their data.

Data analysts work with data engineers to determine and locate appropriate data sources that meet stakeholder requirements, and work with both the data engineer and database administrator to ensure the data analyst has proper access to the needed data sources. The data analyst also works with the data engineer to identify new processes or improve existing processes for collecting data for analysis.

Data Engineer

Data engineers provision and set up data platform technologies that are on-premises and in the cloud. They manage and secure the flow of structured and unstructured data from multiple sources. The data platforms they use can include relational databases, nonrelational databases, data streams, and file stores. Data engineers also ensure that data services securely and seamlessly integrate across data services.

Primary responsibilities of data engineers include the use of on-premises and cloud data services and tools to ingest, egress, and transform data from multiple sources. Data engineers collaborate with business stakeholders to identify and meet data requirements. They design and implement solutions.

While there might be some alignment in the tasks and responsibilities of a data engineer and a database administrator, a data engineer's scope of work goes well beyond looking after a database and the server where it's hosted, and likely doesn't include the overall operational data management.

A data engineer adds tremendous value to both business intelligence and data science projects. When the data engineer brings data together, often described as data wrangling, projects move more quickly because data scientists can focus on their own areas of work.

As a data analyst, you work close with a data engineer in making sure that you can access the variety of structured and unstructured data sources, as they will support you in optimizing data models which are typically served from a modern data warehouse or data lake.

Both database administrators and business intelligence professionals can easily transition to a data engineer role. They just need to learn the tools and technology that are used to process large amounts of data.

Data Scientist

Data scientists perform advanced analytics to extract value from data. Their work can vary from descriptive analytics to predictive analytics. Descriptive analytics evaluate data through a process known as exploratory data analysis (EDA). Predictive analytics are used in machine learning to apply modeling techniques that can detect anomalies or patterns. These are an important part of forecast models.

Descriptive and predictive analytics are just one aspect of data scientists' work. Some data scientists might even work in the realms of deep learning, iteratively experimenting to solve a complex data problem by using customized algorithms.

Anecdotal evidence suggests that most of the work in a data science project is spent on data wrangling and feature engineering. Data scientists can speed up the experimentation process when data engineers use their skills to successfully wrangle data.

On the surface, it might seem that a data scientist and data analyst are far apart in the work they do, but this not the case. A data scientist looks at data to determine the questions that need answers and often will devise a hypothesis or an experiment and turn to the data analyst to assist with the data visualization and reporting.

Database Administrator

A database administrator implements and manages the operational aspects of cloud-native and hybrid data platform solutions built on Microsoft Azure data services and Microsoft SQL Server. They are responsible for the overall availability and consistent performance and optimizations of the database solutions. They work with stakeholders to identify and implement the policies, tools, and processes for data backup and recovery plans.

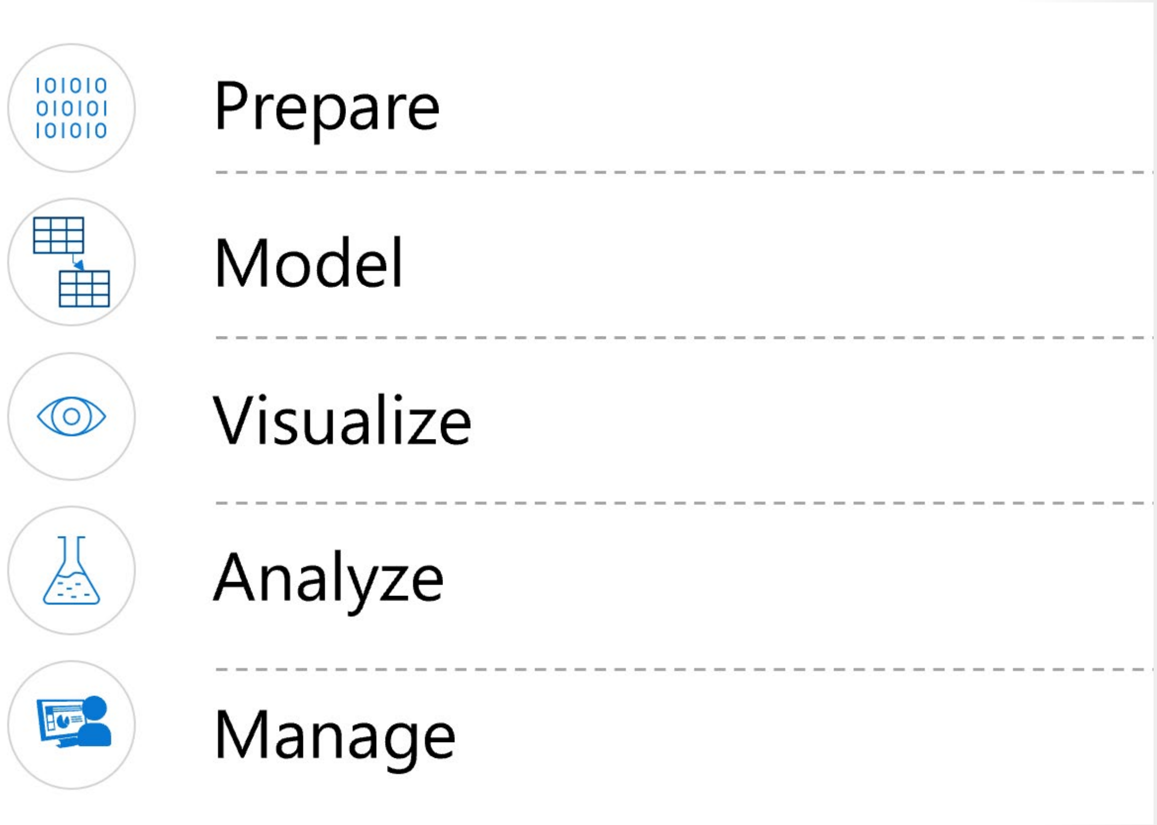
The role of a database administrator is different from the role of a data engineer. A database administrator monitors and manages the overall health of a database and the hardware it resides on, whereas a data engineer is involved in the process of data wrangling, i.e., ingesting, transforming, validating, and cleaning data to meet business needs and requirements.

The database administrator is also responsible for managing the overall security of the data, granting and restricting user access and privileges to the data as determined by business needs and requirements.

Tasks of a data analyst

Now that you've looked at the who and the what, it's time to look at the how. You already know that a data analyst is one of several critical roles in an organization, helping uncover and make sense of information to keep the company balanced and operating efficiently. As such, it's vital that you, as a data analyst, clearly understands your responsibility and the tasks that are performed on a near-daily basis. The skillset of a data analyst is essential in helping organizations gain valuable insights into the expanse of data they have, working closely with others in their respective roles in the organization to help bring to light valuable information.

As such, there are five key areas that you'll engage in during the data analysis process.



Prepare

As a data analyst, you'll probably spend most of your time split between the prepare and model tasks. Bad or incorrect data can have a major impact, resulting in invalid reports, a loss of trust, and can negatively affect business decisions, which lead to loss in revenue, a negative business impact, and more.

Long before a report can be created, data must be prepared. Data preparation is the process of profiling, cleaning, and transforming your data to get it ready to model and visualize.

Data preparation is the process of taking raw data and turning it into information that is trusted and understandable. It involves, among other things, ensuring the integrity of the data, correcting wrong or inaccurate data, identifying missing data, converting data from one structure to another or from one type to another, or even something as simple as making data more readable.

Data preparation also involves understanding "how" you're going to get and connect to the data and the performance implications of the decisions. When connecting to data, decisions need to be made to ensure models and reports meet, and perform to, acknowledged requirements and expectations.

Privacy and security are also important. This can include anonymizing data to avoid oversharing or preventing people from seeing personally identifiable information when it isn't needed. Or, removing that data completely if it doesn't fit in with the story you're trying to shape.

Data preparation can often be a lengthy process, which takes a data analyst through a series of steps and methods of preparing the data to put it in proper context and a state that eliminates poor data quality and allows it to be turned into valuable insights.

Model

Once the data is in a proper state, it's ready to be modeled. Data modeling is the process of determining how your tables are related to each other. This is done by defining, and creating, relationships between the tables. From there, you can enhance the model by defining metrics and adding custom calculations to enrich your data.

Creating an effective and proper data model is a vital and critical step in helping organizations understand and gain valuable insights into the data. An effective data model makes reports more accurate and truthful, allows the data to be explored faster and more efficient, allows the report authoring process to take less time, and makes future report maintenance easier.

It is important to remember that the model is another critical component that has a direct effect on the performance of your report and overall data analysis. A poorly designed model can have a drastically negative impact on the general accuracy and performance of your report. Conversely, a well-designed model with well-prepared data will ensure a properly performant and trusted report. This is even more the case when you are working with data at scale.

From a Power BI perspective, if your report is performing slowly, or your refreshes are taking a long time, you will probably need to revisit the data preparation and modeling tasks to optimize your report.

It should be noted that the process of preparing data and modeling data is an iterative process. Data preparation is the first task in data analysis. Understanding and preparing your data before you model it will make the modeling step much easier.

Visualize

This task is where you get to bring your data to life! The goal of the visualize task is to ultimately solve business problems. A well-designed report should tell a compelling and impactful story about that data, enabling business decision makers to quickly gain needed insights. Using appropriate visualizations and interactions, an effective report guides the reader through the content quickly and efficiently, allowing the reader to follow a narrative into the data.

The reports created during the visualization task help businesses and decision makers understand what that data means so that accurate and vital decisions can be made. Reports drive the overall actions, decisions, and behaviors of an organization, trusting and relying on the information discovered in the data.

The business may communicate that they need all data points on a given report to make decisions. As a data analyst, you should take the time to really understand the problem the business is trying to solve. Are all data point necessary? Too much data can make it difficult to spot the key points. Having a small and concise data story can help to find insights more quickly.

With the built-in AI capabilities in Power BI, data analysts can build powerful reports, without writing any code, that enables users to get insights and answers and find actionable insights. The AI capabilities in Power BI such as the built-in AI visuals enables the discovering of data simply by asking questions, using the Quick Insights feature, or creating machine learning models directly within Power BI.

An important aspect of visualizing data is designing and creating reports for accessibility. As you build reports, it is important to think about those individuals who will be accessing and reading the reports. Reports should be designed with accessibility in mind from the outset so that no special adoptions are needed down the road and no rework is needed.

Many components of your report will help with storytelling. From a color scheme that is complimentary and accessible, fonts, and sizing, to picking the right visuals for what is being displayed, they all come together to tell that story.

Analyze

The Analyze task is the important step of understanding and interpreting the information that is displayed on the report. In your role as a data analyst, you should understand the analytical capabilities of Power BI and use those to find insights, identify patterns and trends, predict outcomes, and then communicate those insights in such a way that everyone can understand.

Advanced analytics enables businesses and organizations to ultimately drive better decisions throughout the business and create actionable insights and meaningful results. With advanced analytics, organizations can drill down into the data to predict future patterns and trends, identify activities and behaviors, and enable businesses to ask the appropriate questions about their data.

In the past, analyzing data was a difficult and intricate process typically carried out and performed by data engineers or data scientists, but today Power BI puts data analysis at your fingertips, simplifying the data analysis process. Users can quickly gain insights into their data using visuals and metrics directly from their desktop and publish those insights to dashboards so that others can find needed information.

This is another area where the AI integrations within Power BI can take your analysis to the next level. Integrations with Azure Machine Learning, cognitive services and those built-in AI visuals help to enrich your data and analysis.

Manage

There are many components in Power BI, including reports, dashboards, workspaces, datasets, and more. As a data analyst, you are responsible for the management of these Power BI assets, overseeing the sharing and distribution of items such as reports and dashboards, and ensuring the security of Power BI assets.

Apps can be a valuable distribution method for your content and allow for easier management for large audiences. This also allows you have custom navigation experiences and link to other assets within your organization to complement your reports.

The management of your content helps to foster collaboration between teams and individuals. Sharing and discovery of your content is important for the right people to get the answers they need. Ensuring items are secure is as important. You want to make sure the right people have access and that you are not leaking data past the correct stakeholders.

Proper management can also help reduce data silos within your organization. Data duplication can make things hard to manage as well as introduce data latency when resources are overused. Power BI helps in reducing data silos with the use of shared datasets. Reuse of data that you have prepared and modeled. For key business data, endorsing a dataset as certified can help to ensure trust in that data.

The management of Power BI assets helps reduce the duplication of efforts and ensures the security of the data.

Lesson Review

In this module, you learned that the role of data analyst is vital to the success of an organization, and the tasks they perform ensure that the business decisions made are based of trusted data. You also learned about the different roles in data and how they work closely with a data analyst to deliver valuable insights into a business's data assets.

Knowledge Check

Question 1

Which data role enables advanced analytics capabilities through reports and visualizations?

- ☐ Data analyst
- ☐ Data scientist
- ☐ Data engineer

Question 2

Which data analyst task has critical performance impact on reporting and data analysis?

- ☐ Analyze
- ☐ Visualize
- ☐ Model

Question 3

What is a key benefit of data analysis?

- ☐ Decisive analytics
- ☐ Informed business decisions
- ☐ Complex reports

Getting Started with Power BI

Introduction

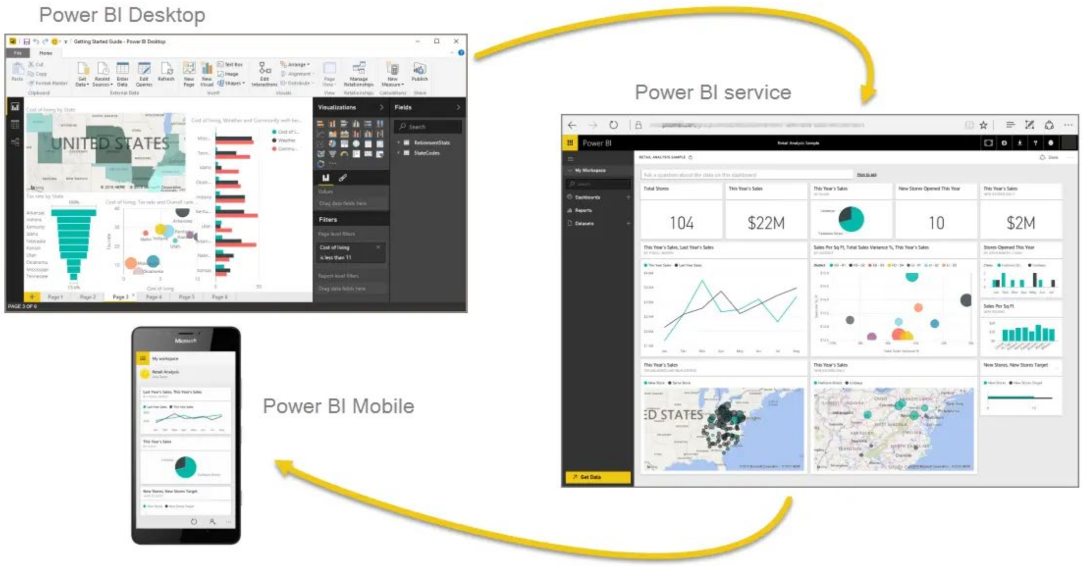
Microsoft Power BI is a collection of software services, apps, and connectors that work together to turn your unrelated sources of data into coherent, visually immersive, and interactive insights. Whether your data is a simple Microsoft Excel workbook, or a collection of cloud-based and on-premises hybrid data warehouses, **Power BI** lets you easily connect to your data sources, visualize (or discover) what's important, and share that with anyone or everyone you want.

Power BI can be simple and fast, capable of creating quick insights from an Excel workbook or a local database. But **Power BI** is also robust and enterprise-grade, ready not only for extensive modeling and real-time analytics, but also for custom development. Therefore, it can be your personal report and visualization tool, but can also serve as the analytics and decision engine behind group projects, divisions, or entire corporations.

If you're a **beginner** with Power BI, this module will get you going. If you're a Power BI **veteran**, this module will tie concepts together and fill in the gaps.

The parts of Power BI

Power BI consists of a Microsoft Windows desktop application called **Power BI Desktop**, an online SaaS (*Software as a Service*) service called the **Power BI service**, and mobile Power BI **apps** that are available on any device, with native mobile BI apps for Windows, iOS, and Android.



These three elements—**Desktop**, the **service**, and **Mobile** apps—are designed to let people create, share, and consume business insights in the way that serves them, or their role, most effectively.

How Power BI matches your role

How you use Power BI might depend on your role on a project or a team. And other people, in other roles, might use Power BI differently, which is just fine.

For example, you might view reports and dashboards in the **Power BI service**, and that might be all you do with Power BI. But your number-crunching, business-report-creating coworker might make extensive use of **Power BI Desktop** (and publish Power BI Desktop reports to the Power BI service, which you then use to view them). And another coworker, in sales, might mainly use her Power BI phone app to monitor progress on her sales quotas and drill into new sales lead details.

You also might use each element of **Power BI** at different times, depending on what you're trying to achieve, or what your role is for a given project or effort.

Perhaps you view inventory and manufacturing progress in a real-time dashboard in the service, and also use **Power BI Desktop** to create reports for your own team about customer engagement statistics. How you use Power BI can depend on which feature or service of Power BI is the best tool for your situation. But each part of Power BI is available to you, which is why it's so flexible and compelling.

We discuss these three elements—**Desktop**, the **service**, and **Mobile** apps—in more detail later. In upcoming units and modules, we'll also create reports in Power BI Desktop, share them in the service, and eventually drill into them on our mobile device.

Download Power BI Desktop

You can download Power BI Desktop from the web or as an app from the Microsoft Store on the Windows tab.

Download Strategy	Link	Notes
Windows Store App	**Windows Store** *(https://aka.ms/pbidesktopstore)*	Will automatically stay updated

Download Strategy	Link	Notes
Download from web	**Download .msi** *(https://go.microsoft.com/fwlink/?LinkID=521662)*	Must manually update periodically

Sign in to Power BI service

Before you can sign in to Power BI, you'll need an account. To get a free trial, go to **app.powerbi.com**[1] and sign up with your email address.

For detailed steps on setting up an account, see **Sign in to Power BI service**[2]

The flow of work in Power BI

A common flow of work in Power BI begins in **Power BI Desktop**, where a report is created. That report is then published to the **Power BI service** and finally shared, so that users of **Power BI Mobile** apps can consume the information.

It doesn't always happen that way, and that's okay. But we'll use that flow to help you learn the different parts of Power BI and how they complement each other.

Okay, now that we have an overview of this module, what Power BI is, and its three main elements, let's take a look at what it's like to use **Power BI**.

Use Power BI

Now that we've introduced the basics of Microsoft Power BI, let's jump into some hands-on experiences and a guided tour.

The activities and analyses that you'll learn with Power BI generally follow a common flow. The **common flow** of activity looks like this:

1. Bring data into Power BI Desktop, and create a report.
2. Publish to the Power BI service, where you can create new visualizations or build dashboards.
3. Share dashboards with others, especially people who are on the go.
4. View and interact with shared dashboards and reports in Power BI Mobile apps.

1 https://go.microsoft.com/fwlink/?linkid=2101313
2 https://docs.microsoft.com/power-bi/consumer/end-user-sign-in

As mentioned earlier, you might spend all your time in the **Power BI service**, viewing visuals and reports that have been created by others. And that's fine. Someone else on your team might spend their time in **Power BI Desktop**, which is fine too. To help you understand the full continuum of Power BI and what it can do, we'll show you all of it. Then you can decide how to use it to your best advantage.

So, let's jump in and step through the experience. Your first order of business is to learn the basic building blocks of Power BI, which will provide a solid basis for turning data into cool reports and visuals.

Building Blocks of Power BI

Everything you do in Microsoft Power BI can be broken down into a few basic **building blocks**. After you understand these building blocks, you can expand on each of them and begin creating elaborate and complex reports. After all, even seemingly complex things are built from basic building blocks. For example, buildings are created with wood, steel, concrete and glass, and cars are made from metal, fabric, and rubber. Of course, buildings and cars can also be basic or elaborate, depending on how those basic building blocks are arranged.

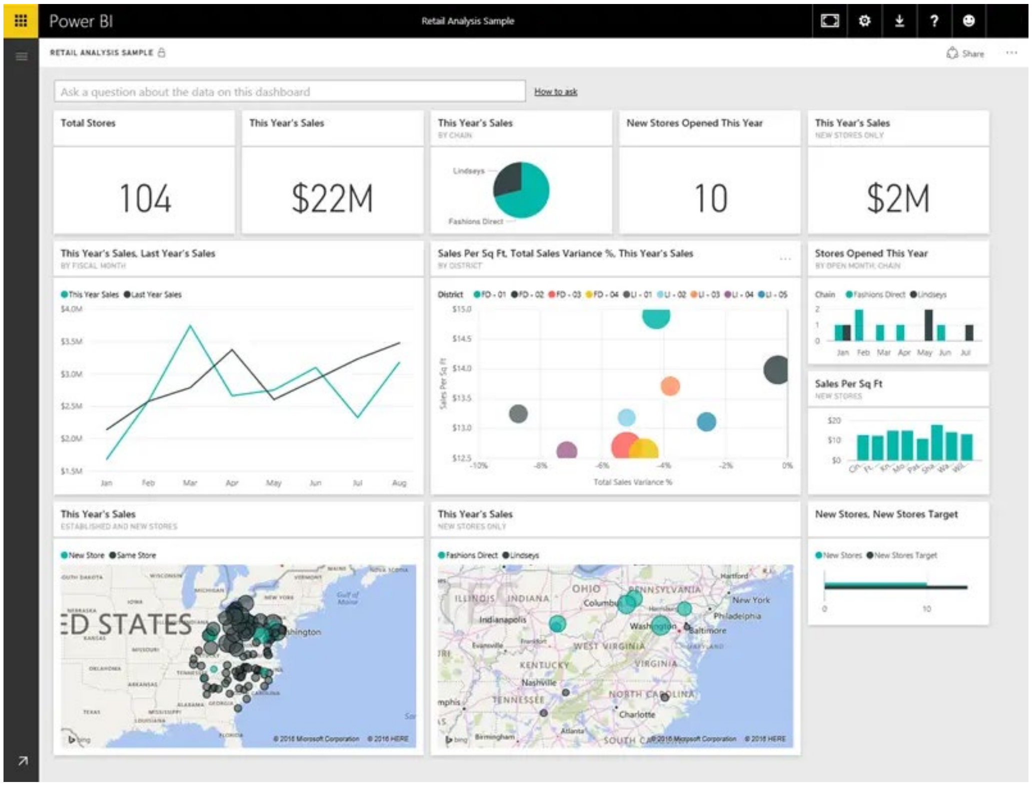
Let's take a look at these basic building blocks, discuss some simple things that can be built with them, and then get a glimpse into how complex things can also be created.

Here are the basic building blocks in Power BI:

- Visualizations
- Datasets
- Reports
- Dashboards
- Tiles

Visualizations

A **visualization** (sometimes also referred to as a **visual**) is a visual representation of data, like a chart, a color-coded map, or other interesting things you can create to represent your data visually. Power BI has all sorts of visualization types, and more are coming all the time. The following image shows a collection of different visualizations that were created in the Power BI service.



Visualizations can be simple, like a single number that represents something significant, or they can be visually complex, like a gradient-colored map that shows voter sentiment about a certain social issue or concern. The goal of a visual is to present data in a way that provides context and insights, both of which would probably be difficult to discern from a raw table of numbers or text.

Datasets

A **dataset** is a collection of data that Power BI uses to create its visualizations.

You can have a simple dataset that's based on a single table from a Microsoft Excel workbook, similar to what's shown in the following image.

C2132 | 2

	Year	Month	Month Name	Calendar Month	Births	Births Per Day	Births (Normalized)
2119	2004	1	January	1/1/2004	2,937	94.7	2842
2120	2004	2	February	2/1/2004	2,824	97.4	2921
2121	2004	3	March	3/1/2004	3,128	100.9	3027
2122	2004	4	April	4/1/2004	2,896	96.5	2896
2123	2004	5	May	5/1/2004	3,008	97.0	2911
2124	2004	6	June	6/1/2004	3,047	101.6	3047
2125	2004	7	July	7/1/2004	2,981	96.2	2885
2126	2004	8	August	8/1/2004	3,079	99.3	2980
2127	2004	9	September	9/1/2004	3,219	107.3	3219
2128	2004	10	October	10/1/2004	3,547	114.4	3433
2129	2004	11	November	11/1/2004	3,365	112.2	3365
2130	2004	12	December	12/1/2004	3,143	101.4	3042
2131	2005	1	January	1/1/2005	2,921	94.2	2827
2132	2005	2	February	2/1/2005	2,699	96.4	2892
2133	2005	3	March	3/1/2005	3,024	97.5	2926
2134	2005	4	April	4/1/2005	3,037	101.2	3037
2135	2005	5	May	5/1/2005	3,231	104.2	3127
2136	2005	6	June	6/1/2005	3,163	105.4	3163
2137	2005	7	July	7/1/2005	3,119	100.6	3018
2138	2005	8	August	8/1/2005	3,156	101.8	3054
2139	2005	9	September	9/1/2005	3,439	114.6	3439

Datasets can also be a combination of many different sources, which you can filter and combine to provide a unique collection of data (a dataset) for use in Power BI.

For example, you can create a dataset from three database fields, one website table, an Excel table, and online results of an email marketing campaign. That unique combination is still considered a single **dataset**, even though it was pulled together from many different sources.

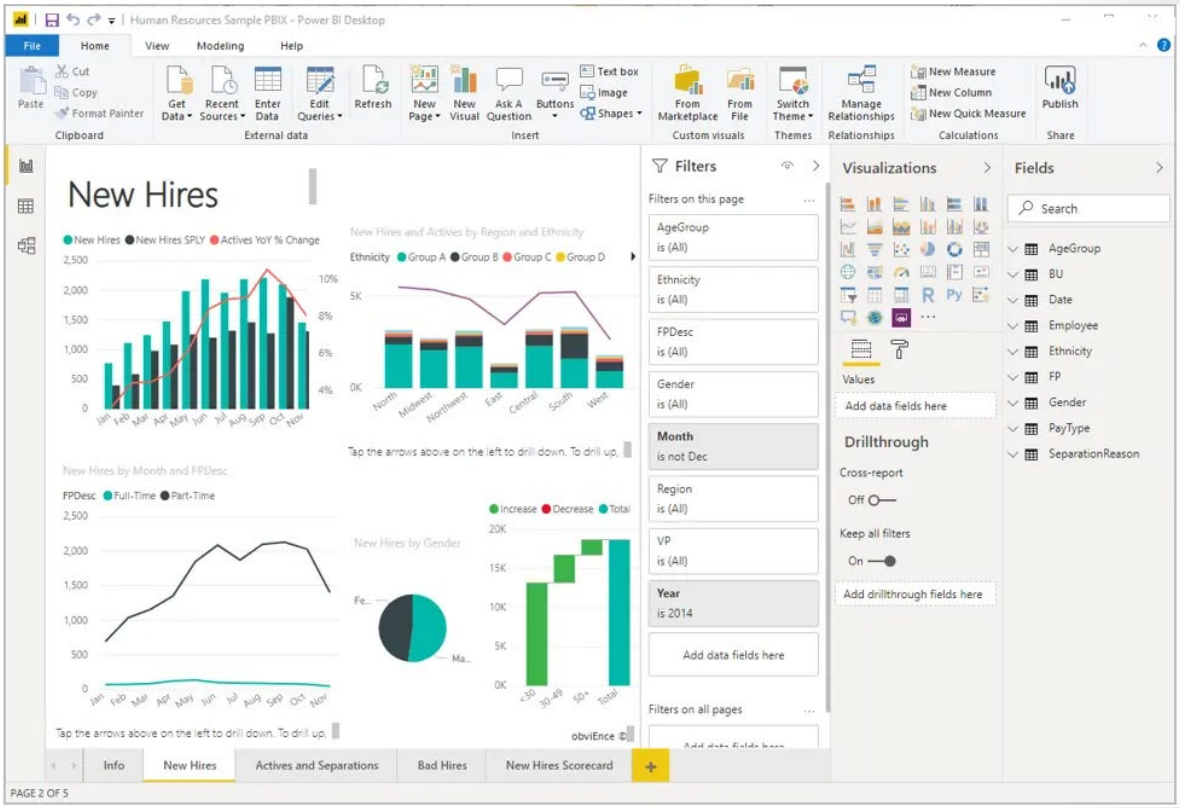
Filtering data before bringing it into Power BI lets you focus on the data that matters to you. For example, you can filter your contact database so that only customers who received emails from the marketing campaign are included in the dataset. You can then create visuals based on that subset (the filtered collection) of customers who were included in the campaign. Filtering helps you focus your data—and your efforts.

An important and enabling part of Power BI is the multitude of data **connectors** that are included. Whether the data you want is in Excel or a Microsoft SQL Server database, in Azure or Oracle, or in a service like Facebook, Salesforce, or MailChimp, Power BI has built-in data connectors that let you easily connect to that data, filter it if necessary, and bring it into your dataset.

After you have a dataset, you can begin creating visualizations that show different portions of it in different ways, and gain insights based on what you see. That's where reports come in.

Reports

In Power BI, a **report** is a collection of visualizations that appear together on one or more pages. Just like any other report you might create for a sales presentation or write for a school assignment, a report in Power BI is a collection of items that are related to each other. The following image shows a **report** in Power BI Desktop—in this case, it's the second page in a five-page report. You can also create reports in the Power BI service.



Reports let you create many visualizations, on multiple pages if necessary, and let you arrange those visualizations in whatever way best tells your story.

You might have a report about quarterly sales, product growth in a particular segment, or migration patterns of polar bears. Whatever your subject, reports let you gather and organize your visualizations onto one page (or more).

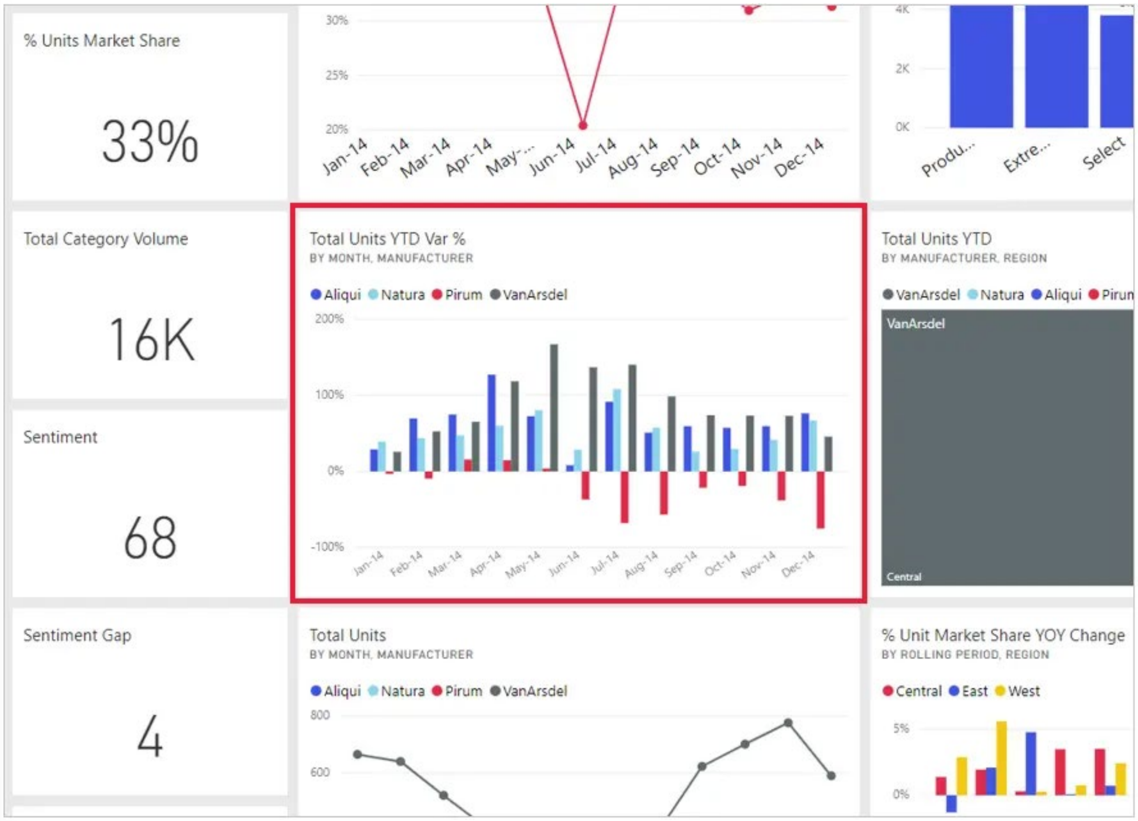
Dashboards

When you're ready to share a single page from a report, or a collection of visualizations, you create a **dashboard**. Much like the dashboard in a car, a Power BI **dashboard** is a collection of visuals from a single page that you can share with others. Often, it's a selected group of visuals that provide quick insight into the data or story you're trying to present.

A dashboard must fit on a single page, often called a canvas (the canvas is the blank backdrop in Power BI Desktop or the service, where you put visualizations). Think of it like the canvas that an artist or painter uses—a workspace where you create, combine, and rework interesting and compelling visuals.
You can share dashboards with other users or groups, who can then interact with your dashboards when they're in the Power BI service or on their mobile device.

Tiles

In Power BI, a **tile** is a single visualization on a report or a dashboard. It's the rectangular box that holds an individual visual. In the following image, you see one tile, which is also surrounded by other tiles.



When you're *creating* a report or a dashboard in Power BI, you can move or arrange tiles however you want. You can make them bigger, change their height or width, and snuggle them up to other tiles.

When you're *viewing*, or *consuming*, a dashboard or report—which means you're not the creator or owner, but the report or dashboard has been shared with you—you can interact with it, but you can't change the size of the tiles or their arrangement.

All together now

Those are the basics of Power BI and its building blocks. Let's take a moment to review.

Power BI is a collection of services, apps, and connectors that lets you connect to your data, wherever it happens to reside, filter it if necessary, and then bring it into Power BI to create compelling visualizations that you can share with others.

Now that you've learned about the handful of basic building blocks of Power BI, it should be clear that you can create datasets that make sense *to you* and create visually compelling reports that tell your story. Stories told with Power BI don't have to be complex, or complicated, to be compelling.

For some people, using a single Excel table in a dataset and then sharing a dashboard with their team will be an incredibly valuable way to use Power BI.

For others, the value of Power BI will be in using real-time Azure SQL Data Warehouse tables that combine with other databases and real-time sources to build a moment-by-moment dataset.

For both groups, the process is the same: create datasets, build compelling visuals, and share them with others. And the result is also the same for both groups: harness your ever-expanding world of data, and turn it into actionable insights.

Whether your data insights require straightforward or complex datasets, Power BI helps you get started quickly and can expand with your needs to be as complex as your world of data requires. And because Power BI is a Microsoft product, you can count on it being robust, extensible, Microsoft Office–friendly, and enterprise-ready.

Now let's see how this works. We'll start by taking a quick look at the Power BI service.

Exercise touring and using Power-BI

As we learned in the previous unit, the common flow of work in Microsoft Power BI is to create a report in Power BI Desktop, publish it to the Power BI service, and then share it with others, so that they can view it in the service or on a mobile app.

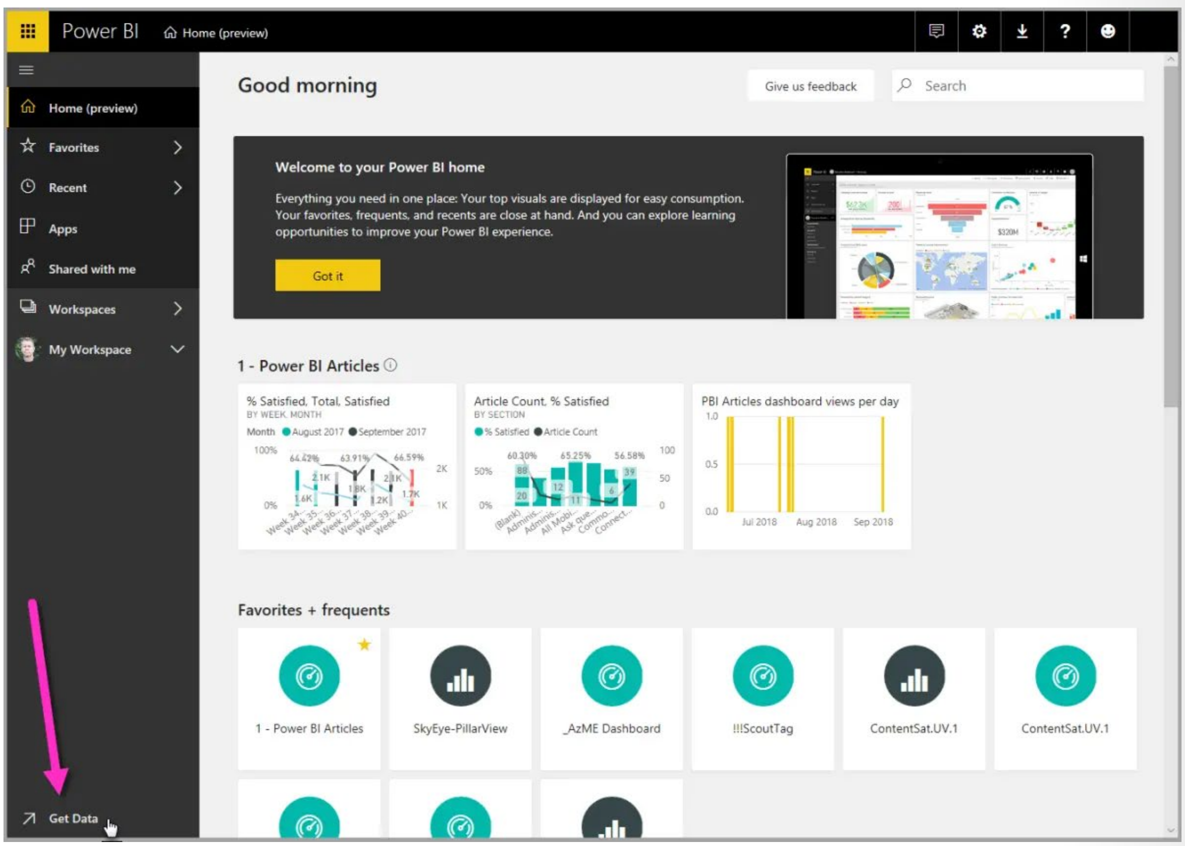
But because some people begin in the Power BI service, let's take a quick look at that first, and learn about an easy and popular way to quickly create visuals in Power BI: *apps*.

An **app** is a collection of preset, ready-made visuals and reports that are shared with an entire organization. Using an app is like microwaving a TV dinner or ordering a fast-food value meal: you just have to press a few buttons or make a few comments, and you're quickly served a collection of entrees designed to go together, all presented in a tidy, ready-to-consume package.

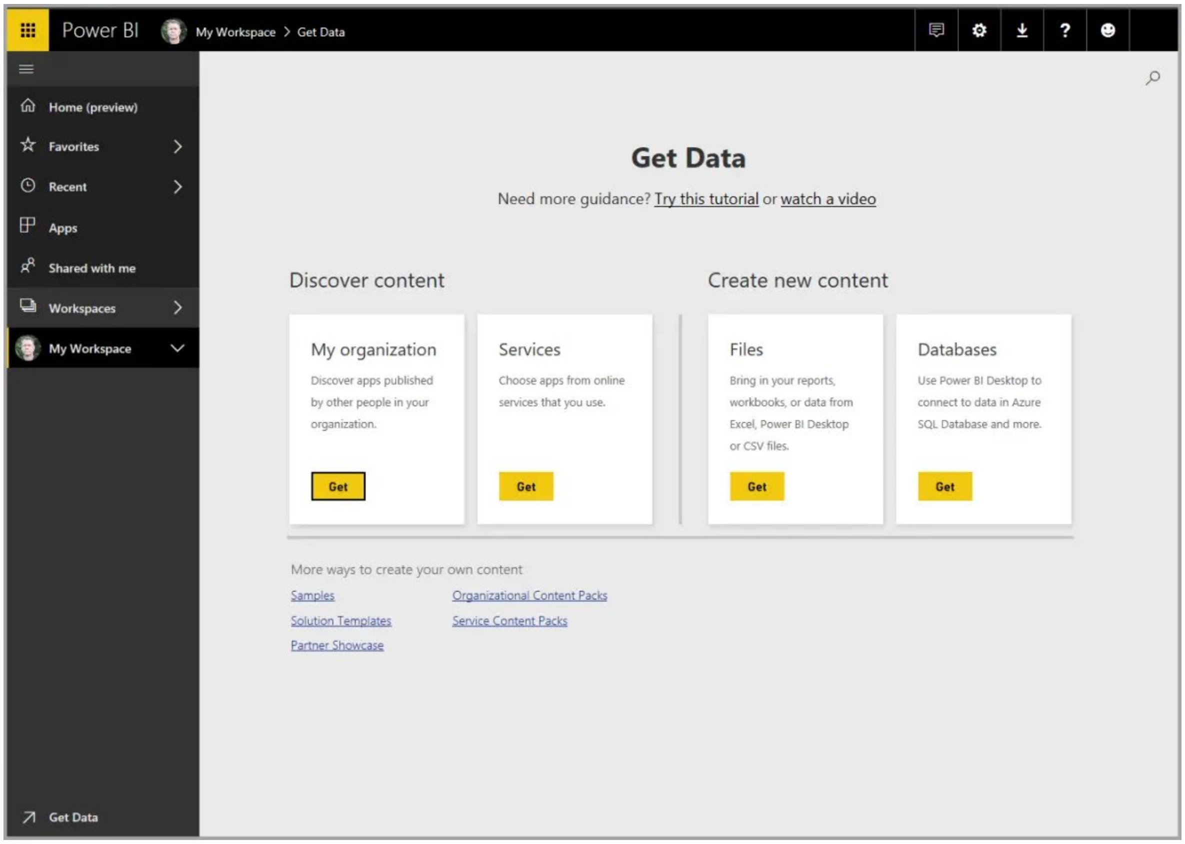
So, let's take a quick look at apps, the service, and how it works. We'll go into more detail about apps (and the service) in upcoming modules, but you can think of this as a taste to whet your appetite.

Create out-of-box dashboards with cloud services

With Power BI, connecting to data is easy. From the Power BI service, you can just select the **Get Data** button in the lower-left corner of the home page.

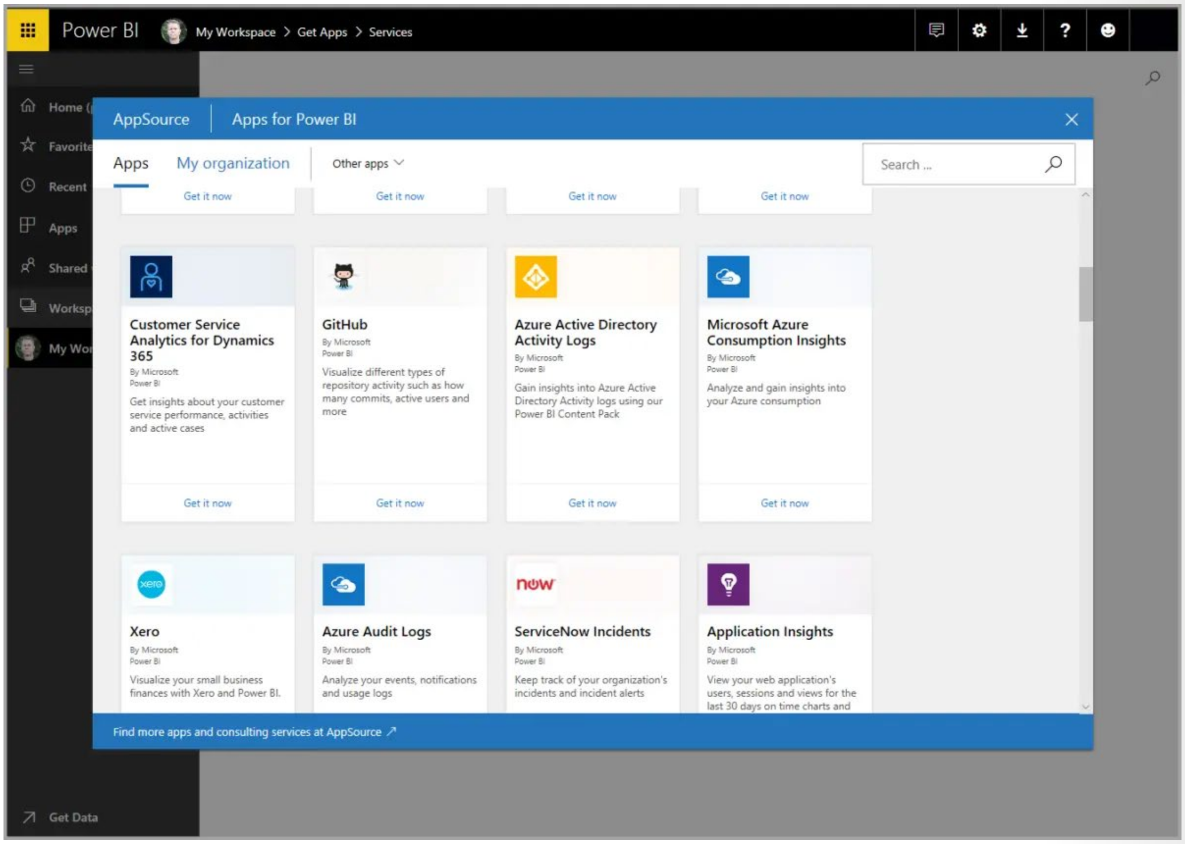


The *canvas* (the area in the center of the Power BI service) shows you the available sources of data in the Power BI service. In addition to common data sources like Microsoft Excel files, databases, or Microsoft Azure data, Power BI can just as easily connect to a whole assortment of **software services** (also called SaaS providers or cloud services): Salesforce, Facebook, Google Analytics, and more.

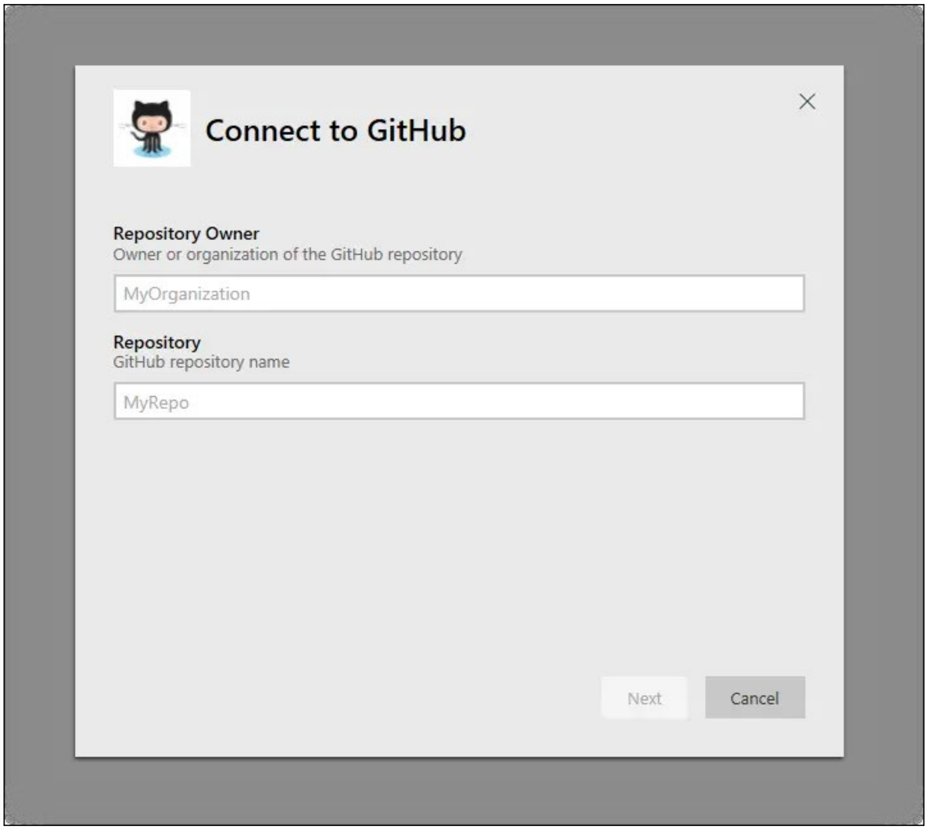


For these software services, the **Power BI service** provides a collection of ready-made visuals that are pre-arranged on dashboards and reports for your organization. This collection of visuals is called an **app**. Apps get you up and running quickly, with data and dashboards that your organization has created for you. For example, when you use the GitHub app, Power BI connects to your GitHub account (after you provide your credentials) and then populates a predefined collection of visuals and dashboards in Power BI.

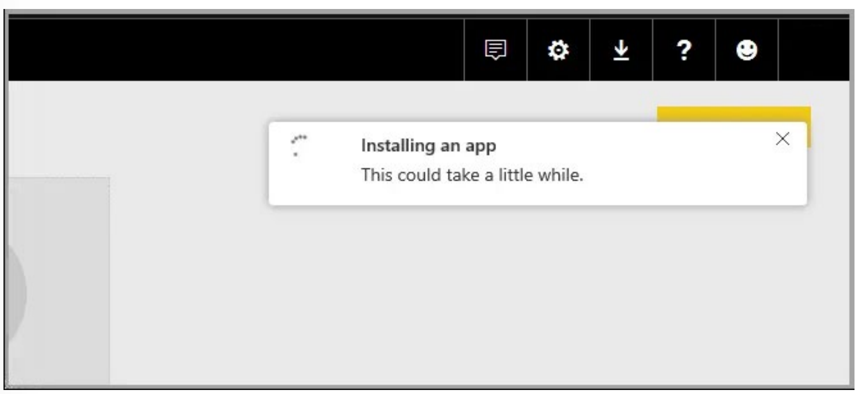
There are apps for all sorts of online services. The following image shows a page of apps that are available for different online services, in alphabetical order. This page is shown when you select the **Get** button in the **Services** box (shown in the previous image). As you can see from the following image, there are many apps to choose from.



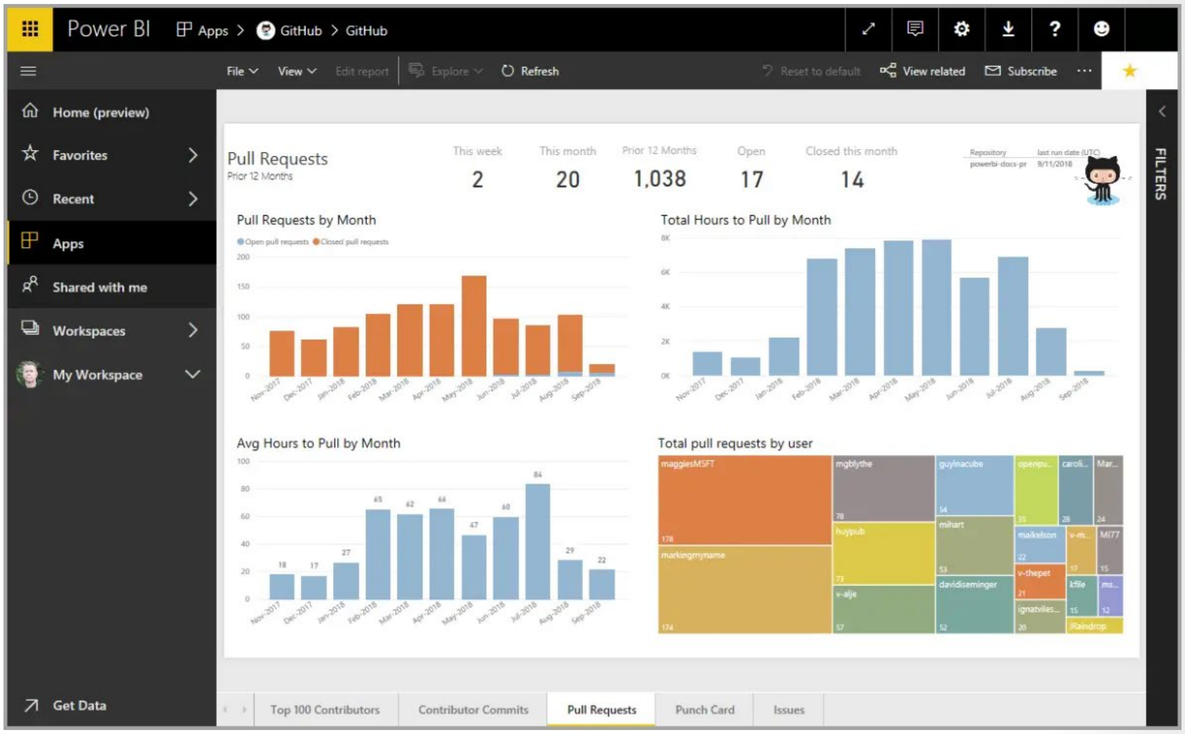
For our purposes, we'll choose **GitHub**. GitHub is an application for online source control. When you select the **Get it now** button in the box for the GitHub app, the **Connect to GitHub** dialog box appears. Note that Github does not support Internet Explorer, so make sure you are working in another browser.



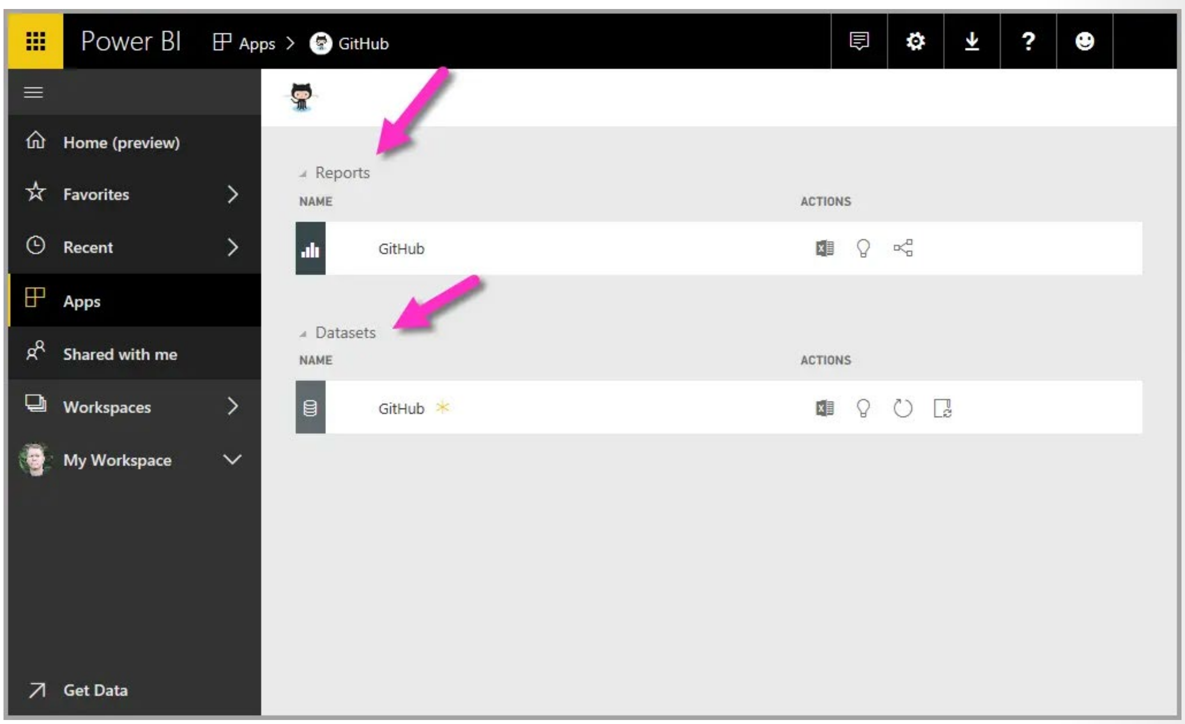
After you enter the information and credentials for the GitHub app, installation of the app begins.



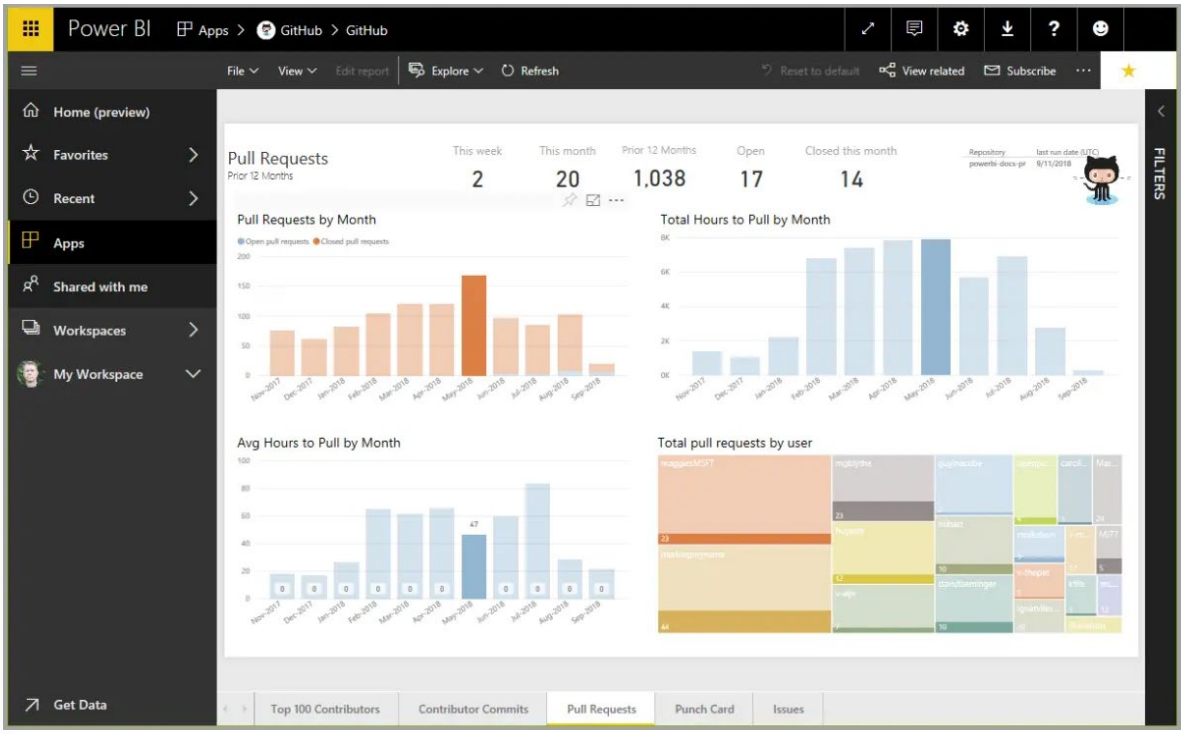
After the data is loaded, the predefined GitHub app dashboard appears.



In addition to the app **dashboard**, the **report** that was generated (as part of the GitHub app) and used to create the dashboard is available, as is the **dataset** (the collection of data pulled from GitHub) that was created during data import and used to create the GitHub report.

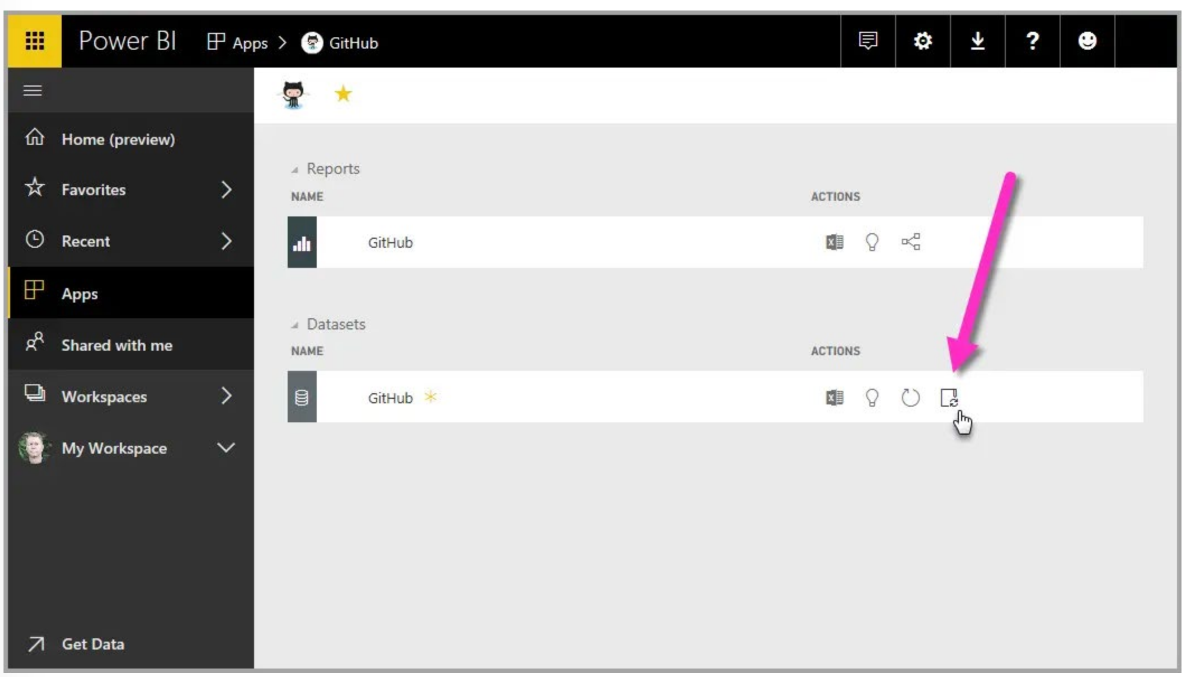


On the dashboard, you can select any of the visuals and interact with them. As you do so, all the other visuals on the page will respond. For example, when the **May 2018** bar is selected in the **Pull Requests (by month)** visual, the other visuals on the page adjust to reflect that selection.

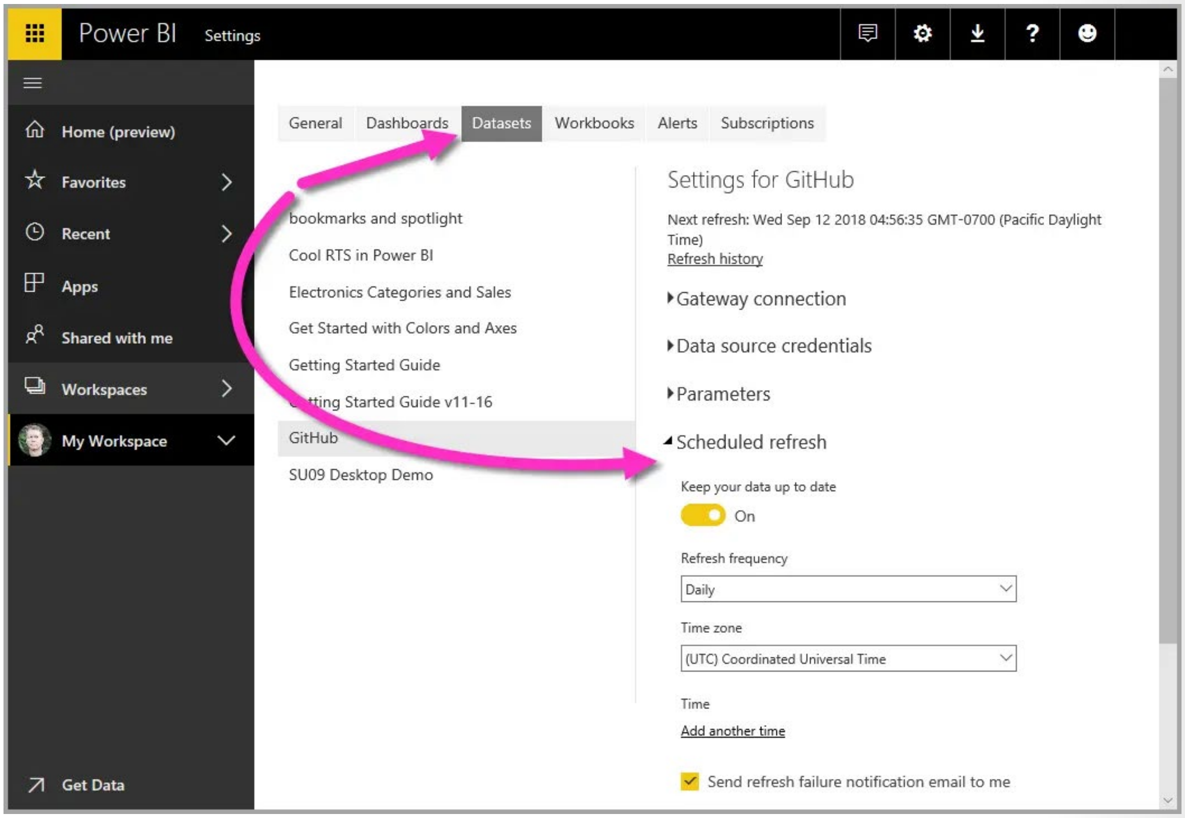


Update data in the Power BI service

You can also choose to **update** the dataset for an app, or other data that you use in Power BI. To set update settings, select the schedule update icon for the dataset to update, and then use the menu that appears. You can also select the update icon (the circle with an arrow) next to the schedule update icon to update the dataset immediately.



The **Datasets** tab is selected on the **Settings** page that appears. In the right pane, select the arrow next to **Scheduled refresh** to expand that section. The **Settings** dialog box appears on the canvas, letting you set the update settings that meet your needs.



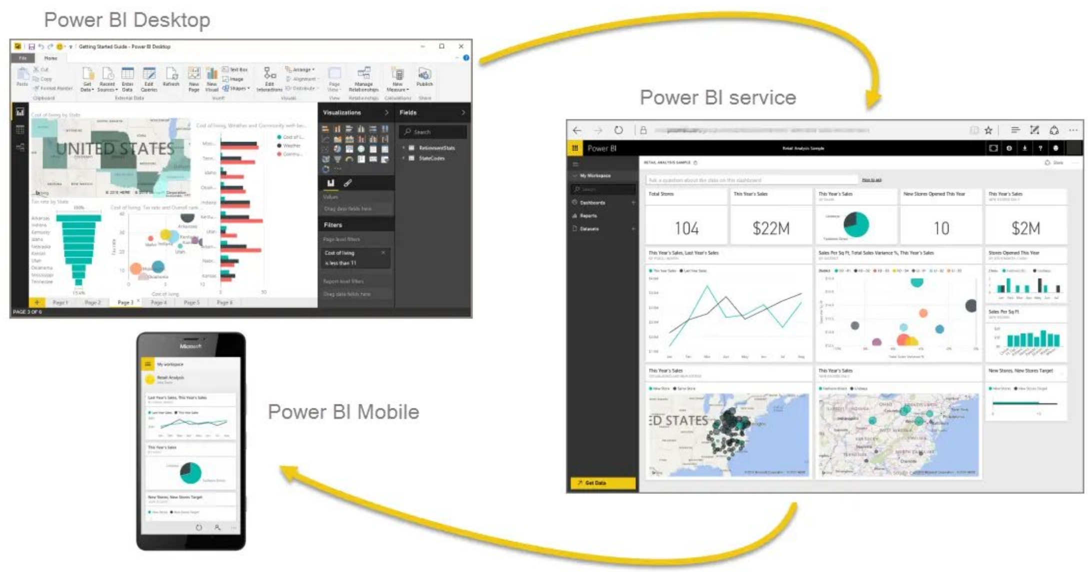
That's enough for our quick look at the Power BI service. There are many more things you can do with the service, and we'll cover these later in this module and in upcoming modules. Also, remember that there are many types of data you can connect to, and all sorts of apps, with more of both coming all the time.

Lesson Review

Let's do a quick review of what we covered in this module.

Microsoft Power BI is a collection of software services, apps, and connectors that work together to turn your data into interactive insights. You can use data from single basic sources, like a Microsoft Excel workbook, or pull in data from multiple databases and cloud sources to create complex datasets and reports. Power BI can be as straightforward as you want or as enterprise-ready as your complex global business requires.

Power BI consists of three main elements—**Power BI Desktop**, the **Power BI service**, and **Power BI Mobile**—which work together to let you create, interact with, share, and consume your data the way you want.



We also discussed the basic building blocks in Power BI:

- **Visualizations** – A visual representation of data, sometimes just called visuals
- **Datasets** – A collection of data that Power BI uses to create visualizations
- **Reports** – A collection of visuals from a dataset, spanning one or more pages
- **Dashboards** – A single-page collection of visuals built from a report
- **Tiles** – A single visualization on a report or dashboard

In the **Power BI service**, we installed an **app** in just a few clicks. That **app**, a ready-made collection of visuals and reports, let us easily connect to a **software service** to populate the app and bring that data to life.

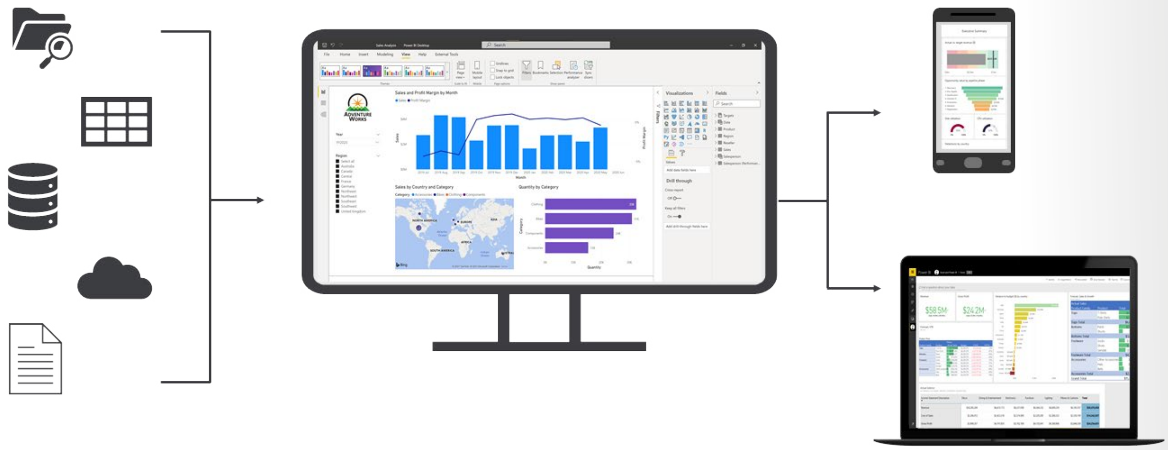
Finally, we set up a **refresh schedule** for our data, so that we know the data will be fresh when we go back to the Power BI service.

Next steps

Congratulations! You've finished the first module of the **learning path** for Power BI. You now have a firm foundation of knowledge for when you move on to the next module, which walks through the steps to create your first report.

We mentioned this before, but it's worth restating: this learning path builds your knowledge by following the common flow of work in Power BI:

- Bring data into **Power BI Desktop**, and create a report.
- **Publish** to the Power BI service, where you create new visualizations or build dashboards.
- **Share** your dashboards with others, especially people who are on the go.
- View and interact with shared dashboards and reports in **Power BI Mobile** apps.



You might not do all that work yourself—some people will only view dashboards that were created by someone else, and they'll just use the service. That's fine, and we'll soon have a module dedicated to showing how you can easily navigate and use the **Power BI service** to view and interact with reports and apps.

But the next module follows the flow of work in Power BI, showing you how to create a report and publish it to the Power BI service. You'll learn how those reports and dashboards are created and how they connected to the data. You might even decide to create a report or dashboard of your own.

See you in the next module!

Knowledge Check

Question 1

What are the building blocks of Power BI?

- ☐ Tiles, dashboards, databases, mobile devices
- ☐ Visual Studio, C#, and JSON files
- ☐ Datasets, Visualizations, Reports, Dashboards, and Tiles

Question 2

What is the common flow of activity in Power BI?

- ☐ Bring data into Power BI Desktop and create a report, share it to the Power BI service, view and interact with reports and dashboards
- ☐ Bring data into Power BI mobile, create a report, then share it to Power BI Desktop.
- ☐ Create a report in the Power BI service, share it to Power BI mobile, interact with it in Power BI Desktop.
- ☐ Create a report in Power BI mobile, share it to the Power BI Desktop, view and interact in the Power BI service.

Question 3

A collection of ready-made visuals, pre-arranged in dashboards and reports is called what?

☐ The canvas

☐ An app

☐ A dataset

☐ Scheduled refresh

Answers

Question 1

Which data role enables advanced analytics capabilities through reports and visualizations?

- ■ Data analyst
- □ Data scientist
- □ Data engineer

Question 2

Which data analyst task has critical performance impact on reporting and data analysis?

- □ Analyze
- □ Visualize
- ■ Model

Question 3

What is a key benefit of data analysis?

- □ Decisive analytics
- ■ Informed business decisions
- □ Complex reports

Question 1

What are the building blocks of Power BI?

- □ Tiles, dashboards, databases, mobile devices
- □ Visual Studio, C#, and JSON files
- ■ Datasets, Visualizations, Reports, Dashboards, and Tiles

Question 2

What is the common flow of activity in Power BI?

- ■ Bring data into Power BI Desktop and create a report, share it to the Power BI service, view and interact with reports and dashboards
- □ Bring data into Power BI mobile, create a report, then share it to Power BI Desktop.
- □ Create a report in the Power BI service, share it to Power BI mobile, interact with it in Power BI Desktop.
- □ Create a report in Power BI mobile, share it to the Power BI Desktop, view and interact in the Power BI service.

Question 3

A collection of ready-made visuals, pre-arranged in dashboards and reports is called what?

- ☐ The canvas
- ■ An app
- ☐ A dataset
- ☐ Scheduled refresh

Module 2 Get Data in Power BI

Get Data from Various Data Sources

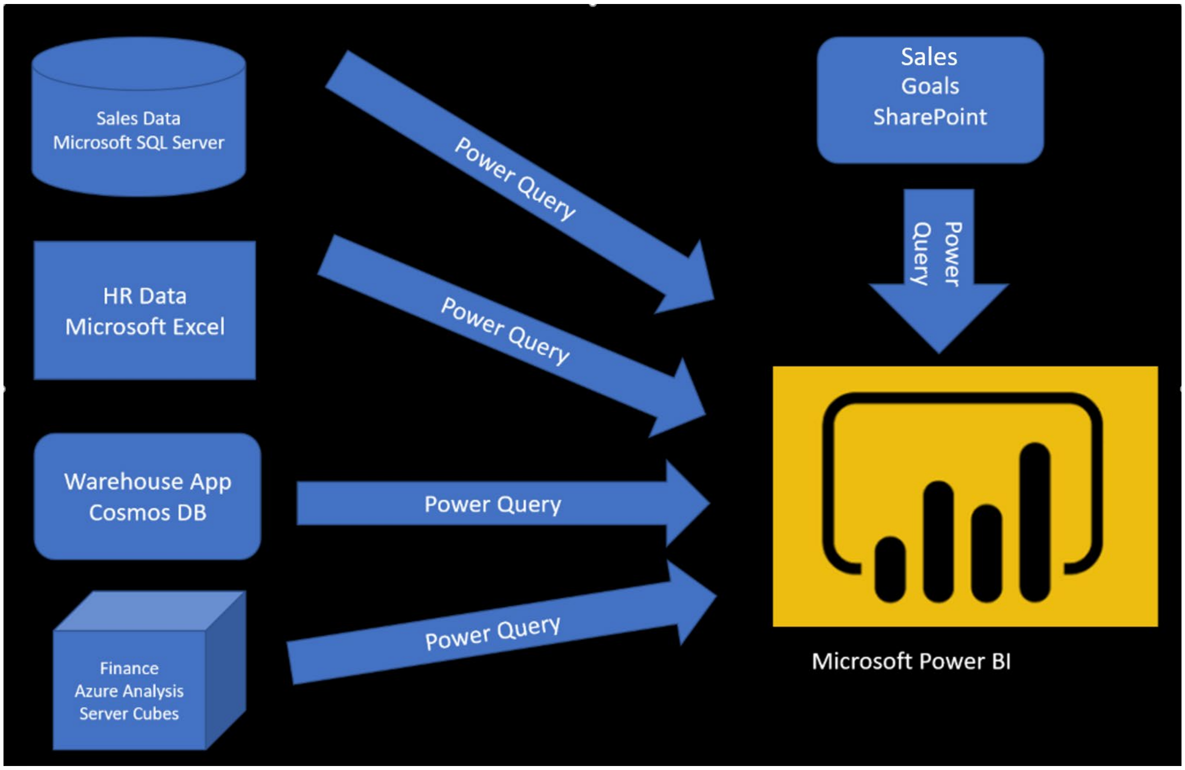
Introduction to getting data

Like most of us, you work for a company where you are required to build Microsoft Power BI reports. The data resides in several different databases and files. These data repositories are different from each other, some are in Microsoft SQL Server, some are in Microsoft Excel, but all the data is related.

In this module's scenario you work for Tailwind Traders. You've been tasked by senior leadership to create a suite of reports that are dependent on data in several different locations. The database that tracks sales transactions is in SQL Server, a relational database contains which customer bought which items and when. It also tracks which employee made the sale, along with the employee name and employee ID. However, that database doesn't contain the employee's hire date, their title, or who their manager is. For that information, you need to access files that Human Resources keeps in Excel. You've been consistently requesting that they use an SQL database, but they haven't yet had the chance to implement it.

When an item ships, the shipment is recorded in the warehousing application, which is new to the company. The developers chose to store data in CosmosDB, as a set of JSON documents.

Tailwind Traders has an application that helps with financial projections, so that they can predict what their sales will be in future months and years, based on past trends. Those projections are stored in Microsoft Azure Analysis Services. Here's a view of the many data sources you are asked to combine data from.



Before you can create reports, you must first extract data from the various data sources. Interacting with SQL Server is different from Excel, so you should learn the nuances of both systems. After you've learned the particulars of each system, you can use Power Query (the query engine used by Power BI and Excel) to help you clean the data, such as renaming columns, replacing values, removing errors, and combining query results. After the data has been cleaned and organized, you are ready to build reports in Power BI. Finally, you will publish your combined dataset and reports to Power BI service (PBIS). From there, other people can use your dataset and build their own reports or they can use the reports that you've already built. Additionally, if someone else built a dataset that you'd like to use, you can build reports from that, too!

This module will focus on the first step, of getting the data from the different data sources and importing it into Power BI by using Power Query.

By the end of this module, you'll be able to:

- Identify and connect to a data source
- Get data from a relational database, such as Microsoft SQL Server
- Get data from a file, such as Microsoft Excel
- Get data from applications
- Get data from Azure Analysis Services
- Select a storage mode
- Fix performance issues
- Resolve data import errors

Get data from flat files

Organizations often export and store data in files. One possible file format is a flat file. A flat file is a type of file that has only one data table and every row of data is in the same structure. The file does not contain hierarchies. Likely, you're familiar with the most common types of flat files, which are comma-separated values (.csv) files, delimited text (.txt) files, and fixed width files. Another type of file would be the output files from different applications, like Microsoft Excel workbooks (.xlsx).

Power BI Desktop allows you to get data from many types of files. You can find a list of the available options when you use the **Get data** feature in Power BI Desktop. The following sections explain how you can import data from an Excel file that is stored on a local computer.

Scenario

The Human Resources (HR) team at Tailwind Traders has prepared a flat file that contains some of your organization's employee data, such as employee name, hire date, position, and manager. They've requested that you build Power BI reports by using this data, and data that is located in several other data sources.

Flat file location

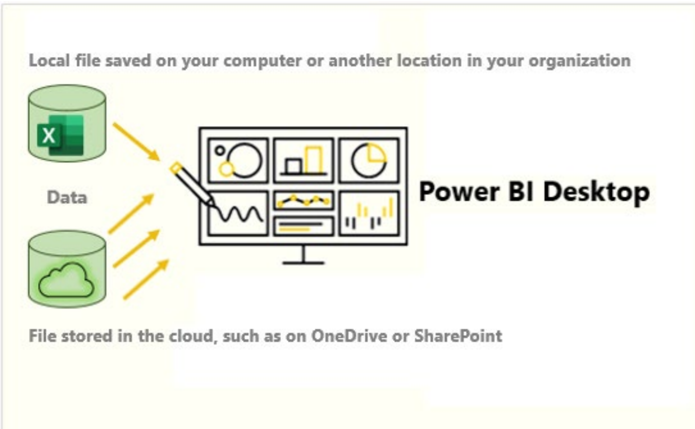
The first step is to determine which file location that you want to use to export and store your data.

Your Excel files might exist in one of the following locations:

- **Local** - You can import data from a local file into Power BI. The file isn't moved into Power BI, and a link doesn't remain to it. Instead, a new dataset is created in Power BI, and data from the Excel file is loaded into it. Accordingly, changes to the original Excel file are not reflected in your Power BI dataset. You can use local data import for data that doesn't change.
- **OneDrive for Business** - You can pull data from OneDrive for Business into Power BI. This method is effective in keeping an Excel file and your dataset, reports, and dashboards in Power BI synchronized. Power BI connects regularly to your file on OneDrive. If any changes are found, your dataset, reports, and dashboards are automatically updated in Power BI.
- **OneDrive - Personal** - You can use data from files on a personal OneDrive account, and get many of the same benefits that you would with OneDrive for Business. However, you'll need to sign in with your personal OneDrive account, and select the **Keep me signed in**

option. Check with your system administrator to determine whether this type of connection is allowed in your organization.

- **SharePoint - Team Sites** - Saving your Power BI Desktop files to SharePoint Team Sites is similar to saving to OneDrive for Business. The main difference is how you connect to the file from Power BI. You can specify a URL or connect to the root folder.

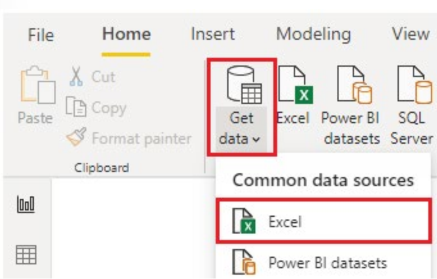


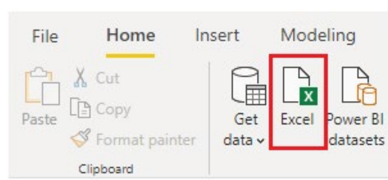
Using a cloud option such as OneDrive or SharePoint Team Sites is the most effective way to keep your file and your dataset, reports, and dashboards in Power BI in-sync. However, if your data does not change regularly, saving files on a local computer is a suitable option.

Connect to data in a file

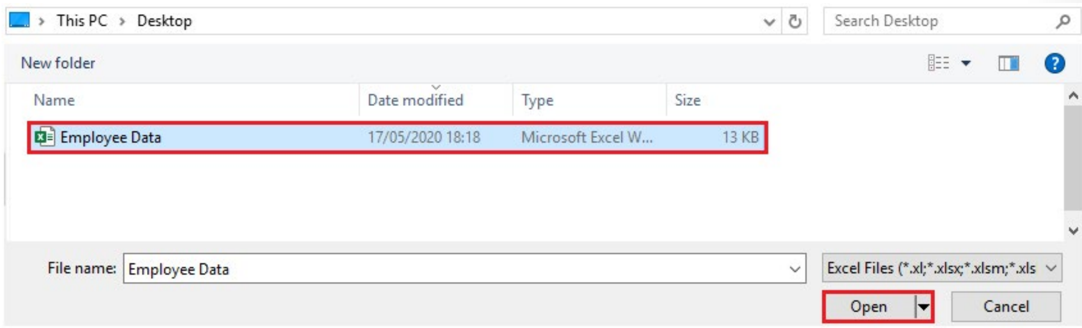
In Power BI, on the **Home** tab, select **Get data**. In the list that displays, select the option that you require, such as **Text/CSV** or **XML**. For this example, you will select **Excel**.

TIP: The **Home** tab contains quick access data source options, such as **Excel**, next to the **Get data** button, as seen in the image below.





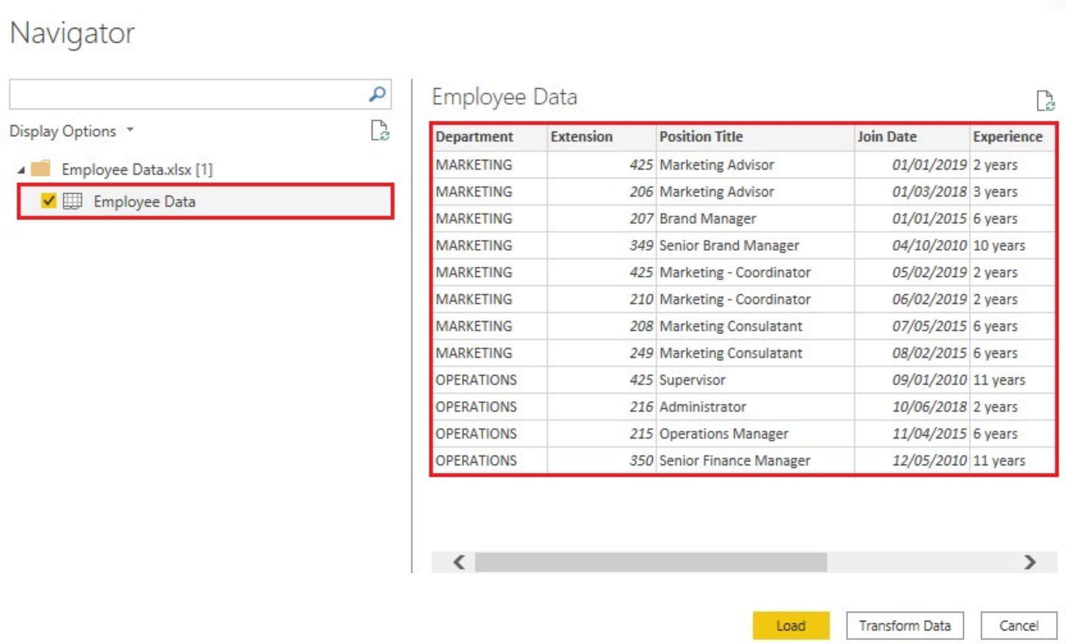
Depending on your selection, you need to find and open your data source. You might be prompted to sign into a service, such as OneDrive, to authenticate your request. In this example, you will open the **Employee Data** Excel workbook that is stored on the Desktop.



Select the file data to import

After the file has connected to Power BI Desktop, the **Navigator** window opens. This window shows you the data that is available in your data source (the Excel file in this example). You can select a table or entity to preview its contents, to ensure that the correct data is loaded into the Power BI model.

Select the check box(es) of the table(s) that you want to bring in to Power BI. This selection activates the **Load** and **Transform Data** buttons as shown in the following image.



Department	Extension	Position Title	Join Date	Experience
MARKETING	425	Marketing Advisor	01/01/2019	2 years
MARKETING	206	Marketing Advisor	01/03/2018	3 years
MARKETING	207	Brand Manager	01/01/2015	6 years
MARKETING	349	Senior Brand Manager	04/10/2010	10 years
MARKETING	425	Marketing - Coordinator	05/02/2019	2 years
MARKETING	210	Marketing - Coordinator	06/02/2019	2 years
MARKETING	208	Marketing Consulatant	07/05/2015	6 years
MARKETING	249	Marketing Consulatant	08/02/2015	6 years
OPERATIONS	425	Supervisor	09/01/2010	11 years
OPERATIONS	216	Administrator	10/06/2018	2 years
OPERATIONS	215	Operations Manager	11/04/2015	6 years
OPERATIONS	350	Senior Finance Manager	12/05/2010	11 years

You now have the option to select the **Load** button to automatically load your data into the Power BI model or select the **Transform Data** button to launch the Power Query Editor, where you can review and clean your data before loading it into the Power BI model.

We often recommend that you transform data, but that process will be discussed later in this module. For this example, you can select **Load**.

Change the source file

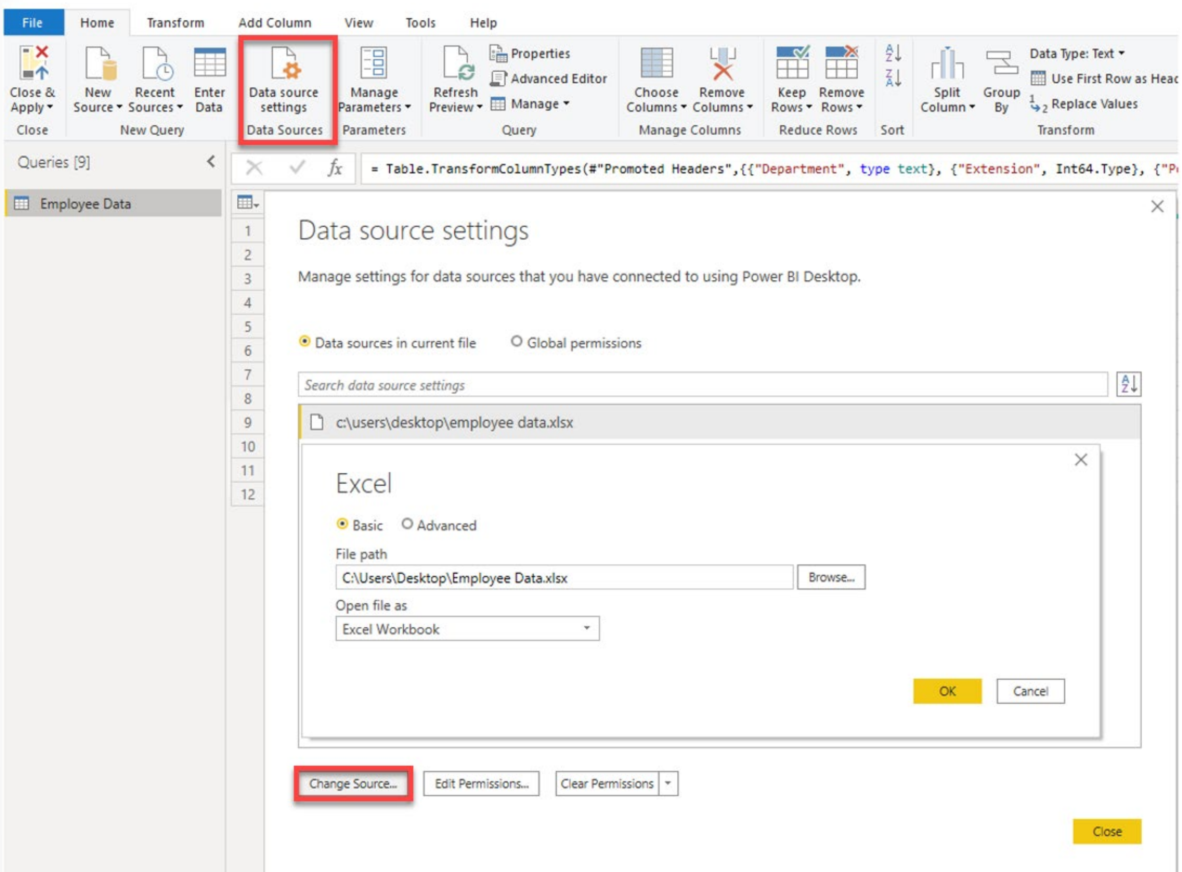
You might have to change the location of a source file for a data source during development, or if a file storage location changes. To keep your reports up to date, you'll need to update your file connection paths in Power BI.

Power Query provides a number of ways for you to accomplish this task, so that you can make this type of change when needed.

1. Data source settings
2. Query settings
3. Advanced Editor

WARNING: If you are changing a file path, make sure that you reconnect to the same file with the same file structure. Any structural changes to a file, such as deleting or renaming columns in the source file, will break the reporting model.

For example, try changing the data source file path in the data source settings. Select **Data source settings** in Power Query. In the **Data source settings** window, select your file and then select **Change Source**. Update the **File path** or use the **Browse** option to locate your file, select **OK**, and then select **Close**.



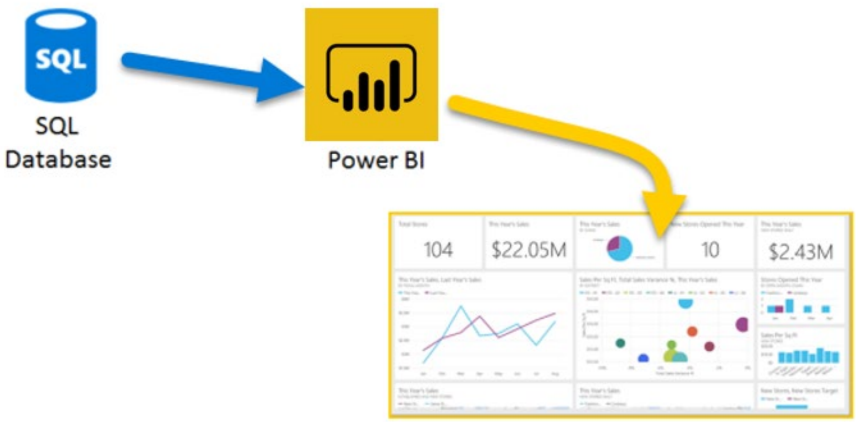
Get data from relational data sources

If your organization uses a relational database to record its sales transactions, you can use Power BI Desktop to establish a connection to your organization's relational database, rather than getting data from individual flat files.

Connecting Power BI to your database will help you to monitor the progress of your business and identify trends, so you can forecast sales figures, plan budgets and set performance indicators and targets. Power BI Desktop can connect to many relational databases that are either in the cloud or on-premises.

Scenario

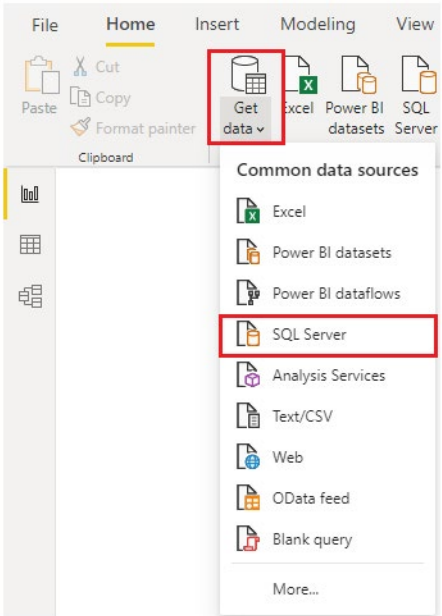
The Sales team at Tailwind Traders have requested that you connect to the organization's on-premises SQL Server database and get the sales data into Power BI Desktop so you can build sales reports.



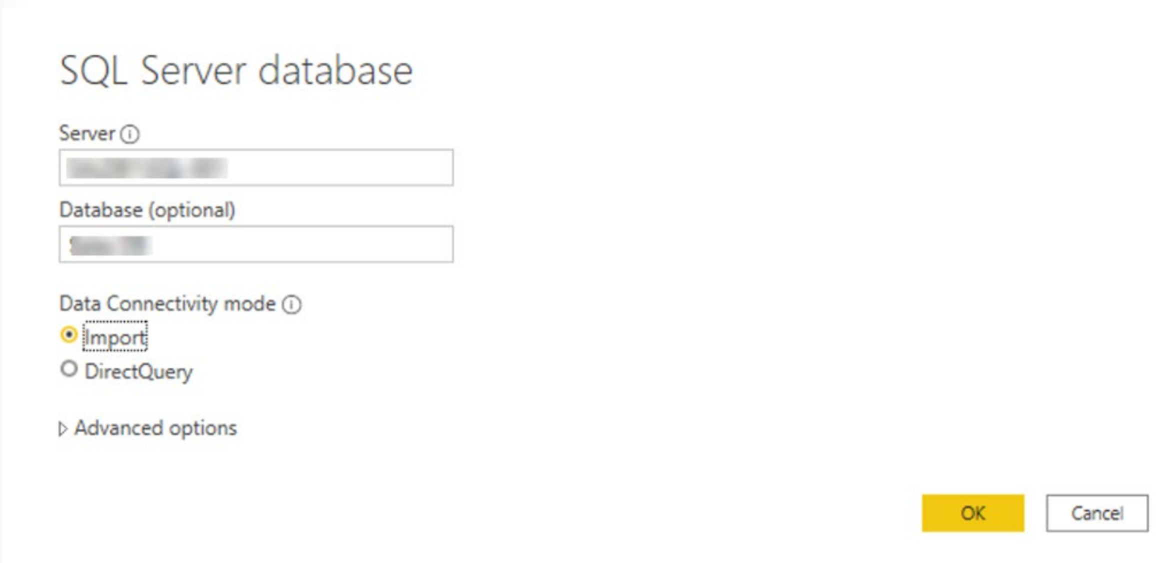
Connect to data in a relational database

You can use the **Get data** feature in Power BI Desktop and select the applicable option for your relational database. For this example, you would select the **SQL Server** option, as shown in the following screenshot.

TIP: Next to the **Get Data** button are quick access data source options, such as **SQL Server**.



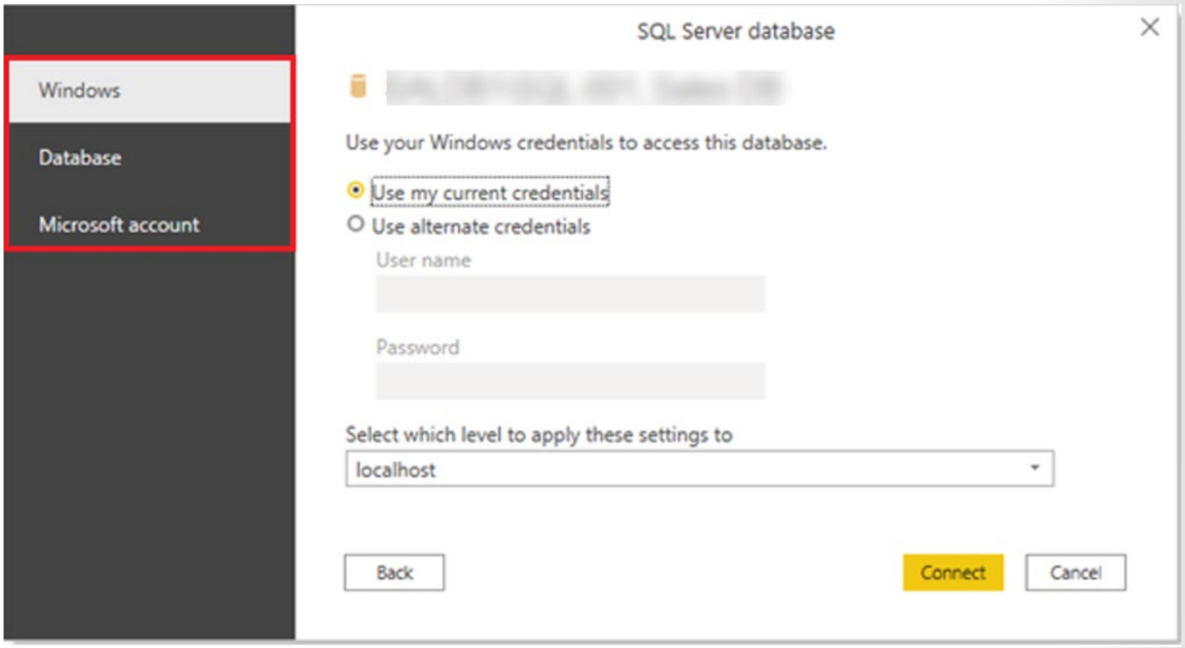
Your next step is to enter your database server name and a database name in the **SQL Server database** window. The two options in data connectivity mode are: **Import** (selected by default, recommended) and **DirectQuery**. Mostly, you select **Import**. Other advanced options are also available in the **SQL Server database** window, but you can ignore them for now.



After you have added your server and database names, you will be prompted to sign in with a username and password. You will have three sign-in options:

- **Windows** - Use your Windows account (Azure Active Directory credentials).
- **Database** - Use your database credentials. For instance, SQL Server has its own sign-in and authentication system that is sometimes used. If the database administrator gave you a unique sign-in to the database, you might need to enter those credentials on the **Database** tab.
- **Microsoft account** - Use your Microsoft account credentials. This option is often used for Azure services.

Select a sign-in option, enter your username and password, and then select **Connect**.

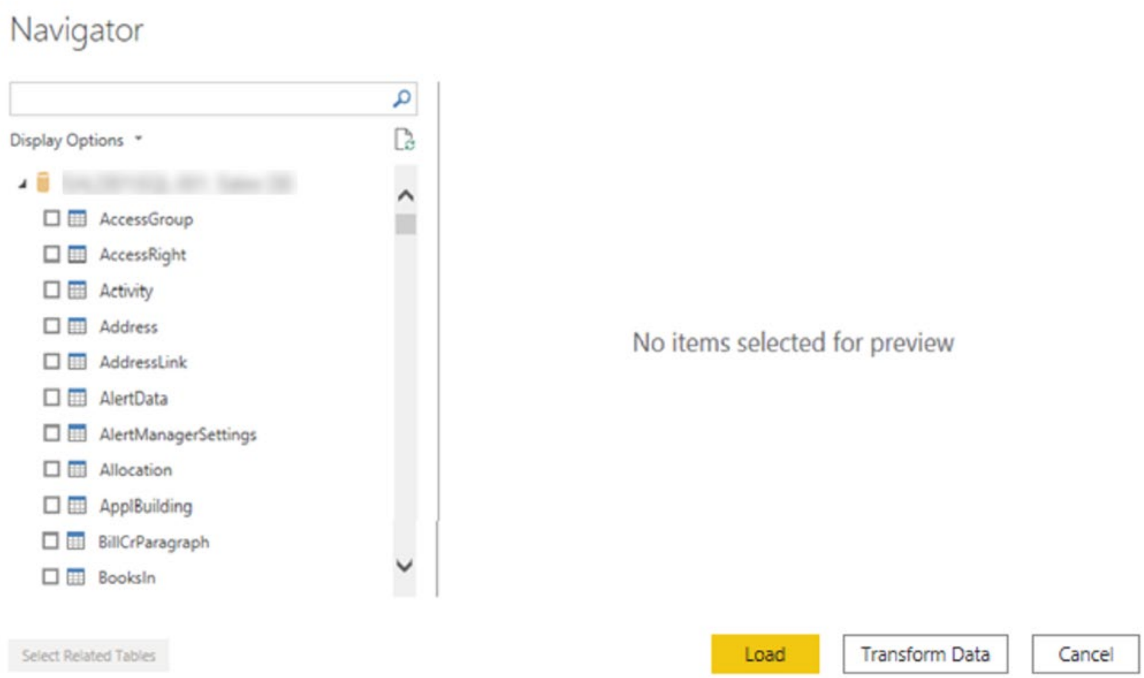


Select data to import

After the database has been connected to Power BI Desktop, the **Navigator** window displays the data that is available in your data source (the SQL database in this example). You can select a table or entity to preview its contents and make sure that the correct data will be loaded into the Power BI model.

Select the check box(es) of the table(s) that you want to bring in to Power BI Desktop, and then select either the **Load** or **Transform Data** option.

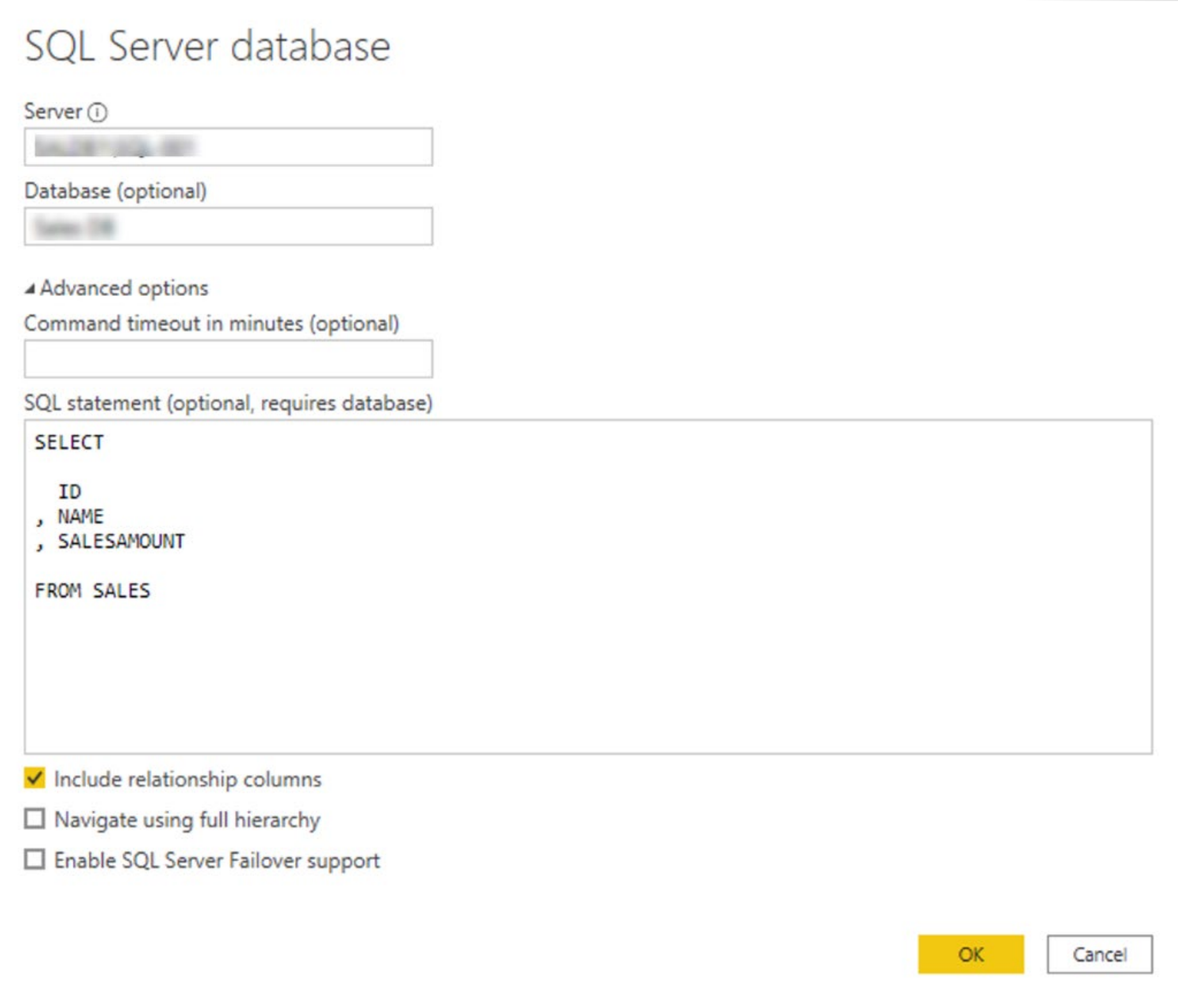
- **Load** - Automatically load your data into a Power BI model in its current state.
- **Transform Data** - Open your data in Microsoft Power Query, where you can perform actions such as deleting unnecessary rows or columns, grouping your data, removing errors, and many other data quality tasks.



Import data by writing an SQL query

Another way you can import data is to write an SQL query to specify only the tables and columns that you need.

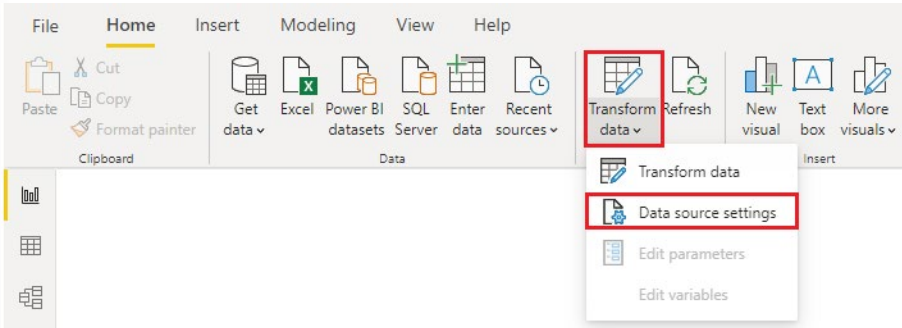
To write your SQL query, on the **SQL Server database** window, enter your server and database names, and then select the arrow next to **Advanced options** to expand this section and view your options. In the **SQL statement** box, write your query statement, and then select **OK**. In this example, you will use the **Select** SQL statement to load the ID, NAME and SALESAMOUNT columns **from** the SALES table.



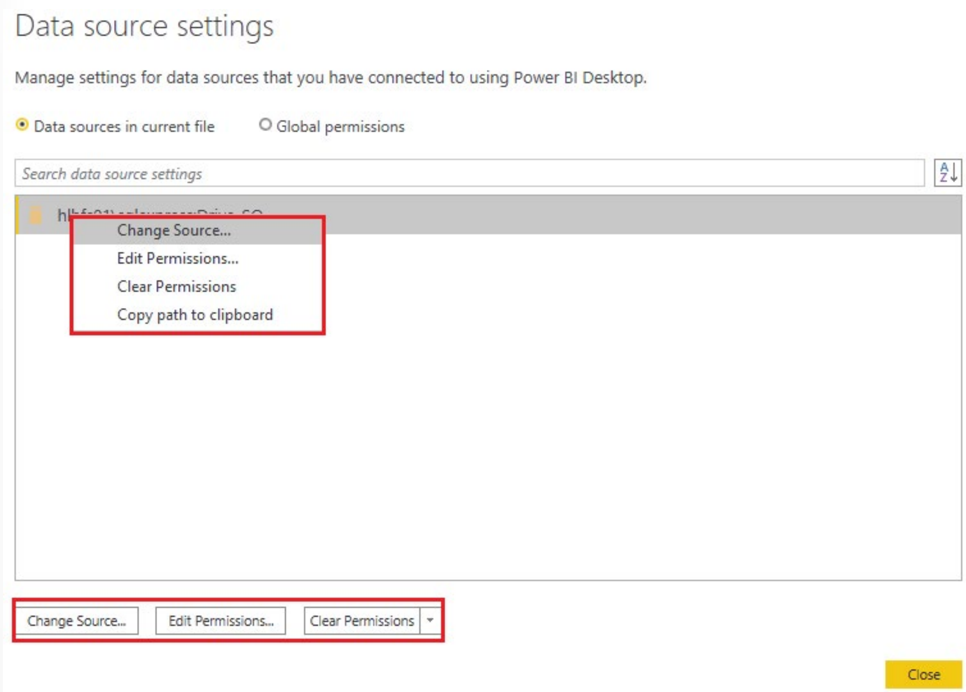
Change data source settings

After you create a data source connection and load data into Power BI Desktop, you can return and change your connection settings at any time. This action is often required due to a security policy within the organization, for example, when the password needs to be updated every 90 days. You can change the data source, edit permissions or clear permissions.

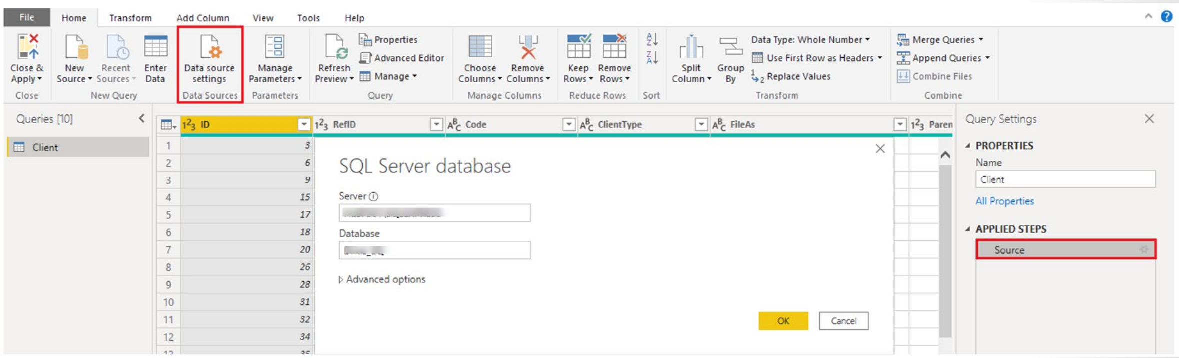
On the **Home** tab, select **Transform data**, and then select the **Data source settings** option.



From the list of data sources that displays, select the data source that you want to update. Then, you can right-click that data source to view the available update options or you can use the update option buttons on the lower left of the window. Select the update option that you need, change the settings as required, and then apply your changes.



You can also change your data source settings from within Power Query. Select the table, and then select the **Data source settings** option on the **Home** ribbon. Alternatively, you can go to the **Query Settings** panel on the right side of the screen and select the settings icon next to Source (or double Select Source). In the window that displays, update the server and database details, and then select **OK**.



After you have made the changes, select **Close and Apply** to apply those changes to your data source settings.

Write an SQL statement

As previously mentioned, you can import data into your Power BI model by using an SQL query. SQL stands for Structured Query Language and is a standardized programming language that is used to manage relational databases and perform various data management operations.

Consider the scenario where your database has a large table that is comprised of sales data over several years. Sales data from 2009 is not relevant to the report that you are creating. This situation is where SQL is beneficial because it allows you to load only the required set of data by specifying exact columns and rows in your SQL statement and then importing them into your data model. You can also join different tables, run specific calculations, create logical statements, and filter data in your SQL query.

The following example shows a simple query where the ID, NAME and SALESAMOUNT are selected from the SALES table.

The SQL query starts with a **Select** statement, which allows you to choose the specific fields that you want to pull from your database. In this example, you want to load the ID, NAME, and SALESAMOUNT columns.

```
SELECT
ID
, NAME
, SALESAMOUNT
FROM
```

FROM specifies the name of the table that you want to pull the data from. In this case, it's the SALES table. The following example is the full SQL query:

```
SELECT
ID
, NAME
, SALESAMOUNT
FROM
SALES
```

When using an SQL query to import data, try to avoid using the wildcard character (*) in your query. If you use the wildcard character (*) in your SELECT statement, you import all columns that you don't need from the specified table.

The following example shows the query using the wildcard character.

```
SELECT *
FROM
SALES
```

The wildcard character (*) will import all columns within the **Sales** table. This method is not recommended because it will lead to redundant data in your data model, which will cause performance issues and require additional steps to normalize your data for reporting.

All queries should also have a **WHERE** clause. This clause will filter the rows to pick only filtered records that you want. In this example, if you want to get recent sales data after Jan 1, 2020, add a **WHERE** clause. The evolved query would look like the following example.

```
SELECT
ID
, NAME
, SALESAMOUNT
FROM
SALES
WHERE
OrderDate >= '1/1/2020'
```

It is a best practice to avoid doing this directly in Power BI. Instead, consider writing a query like this in a view. A view is an object in a relational database, similar to a table. Views have rows and columns, and can contain almost every operator in the SQL language. If Power BI uses a view, when it retrieves data, it participates in query folding, a feature of Power Query. Query folding will be explained later, but in short, Power Query will optimize data retrieval according to how the data is being used later.

Get data from NoSQL

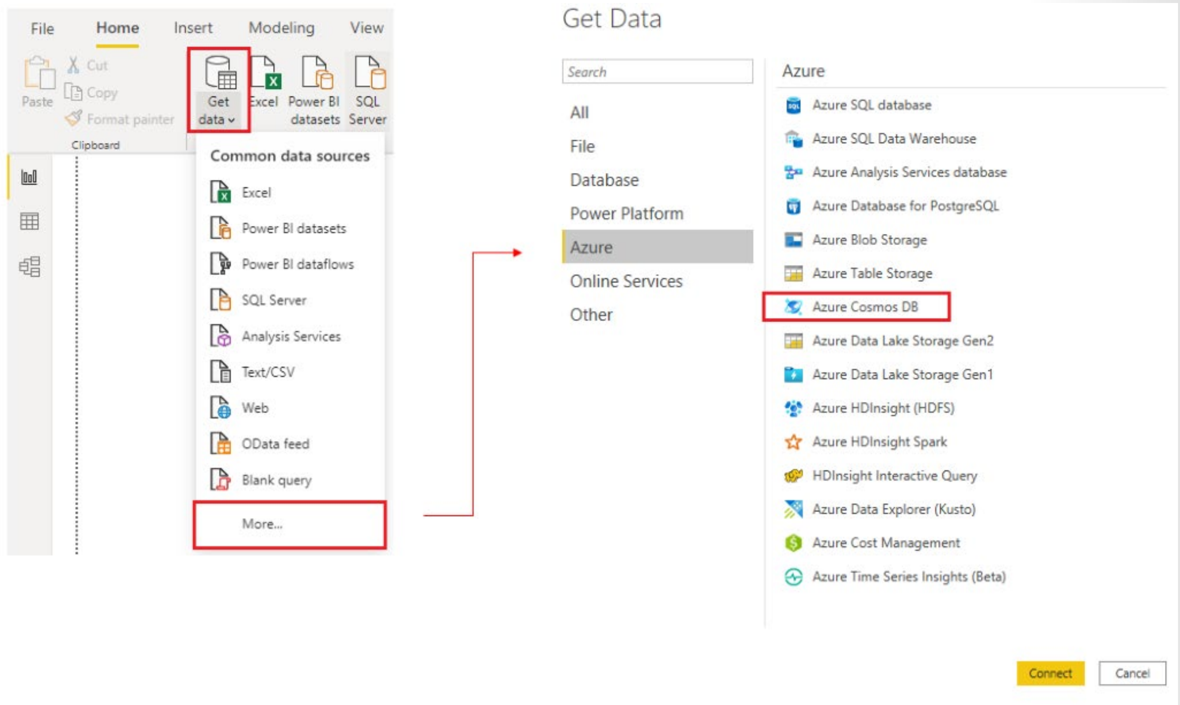
Some organization don't use a relational database but instead use a *NoSQL* database. A NoSQL database (also referred to as non-SQL, not only SQL or *non-relational*) is a flexible type of database that does not use tables to store data.

Scenario

Software developers at Tailwind Traders created an application to manage shipping and tracking products from their warehouses that uses CosmosDB, a NoSQL database, as the data repository. This application uses Cosmos DB to store JSON documents, which are open standard file formats that are primarily used to transmit data between a server and web application. You need to import this data into a Power BI data model for reporting.

Connect to a NoSQL database (Azure Cosmos DB)

In this scenario, you will use the **Get data** feature in Power BI Desktop. However, this time you will select the **More**... option to locate and connect to the type of database that you use. In this example, you will select the **Azure** category, select **Azure Cosmos DB**,and then select **Connect**.



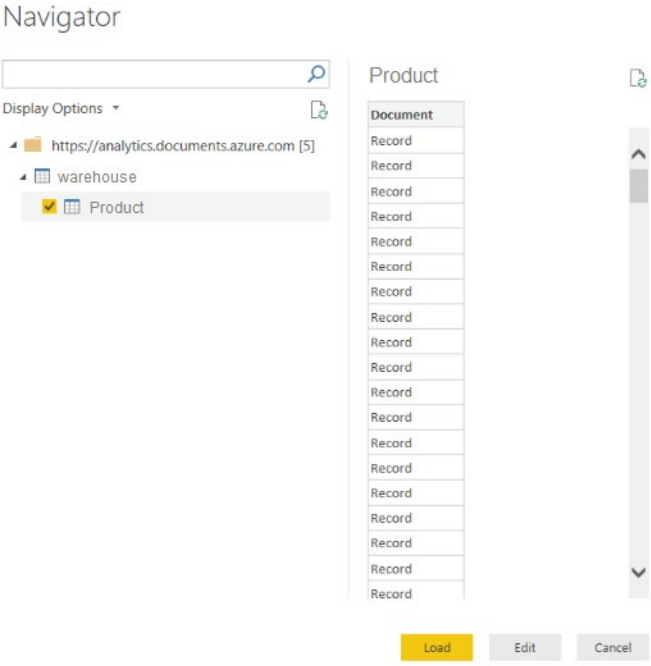
On the **Preview Connector** window, select **Continue** and then enter your database credentials. In this example, on the **Azure Cosmos DB** window, you can enter the database details. You can specify the Azure Cosmos DB account endpoint URL that you want to get the data from (you can get the URL from the **Keys** blade of your Azure portal). Alternatively, you can enter the database name, collection name or use the navigator to select the database and collection to identify the data source.

If you are connecting to an endpoint for the first time, as you are in this example, make sure that you enter your account key. You can find this key in the **Primary Key** box in the **Read-only Keys** blade of your Azure portal.

Import a JSON file

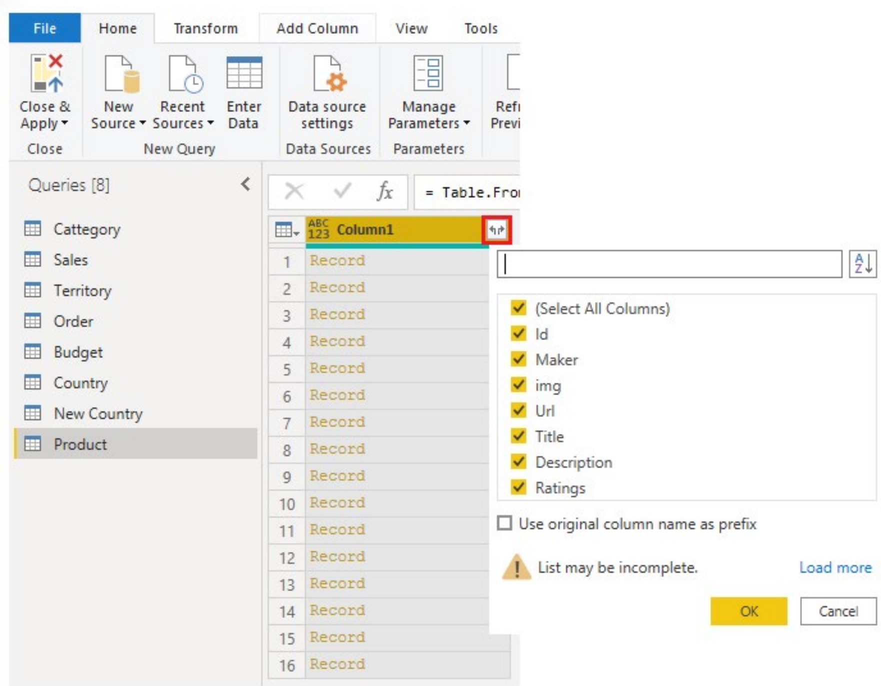
JSON type records must be extracted and normalized before you can report on them, so you need to transform the data before loading it into Power BI Desktop.

After you have connected to the database account, the **Navigator** window opens, showing a list of databases under that account. Select the table that you want to import. In this example, you will select the Product table. The preview pane only shows **Record** items because all records in the document are represented as a Record type in Power BI.

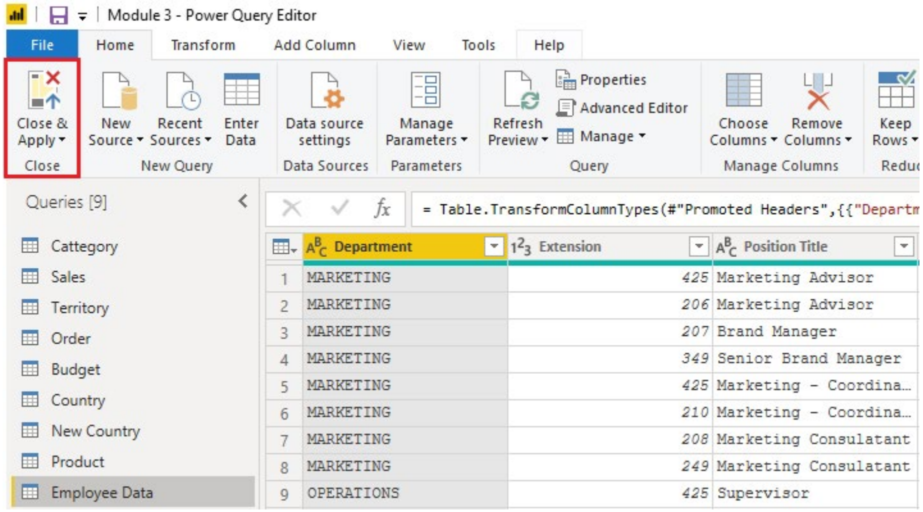


Select the **Edit** button to open the records in Power Query.

In Power Query, select the **Expander** button to the right side of the **Column1** header, which will display the context menu with a list of fields. Select the fields that you want to load into Power BI Desktop, clear the **Use original column name as prefix** checkbox, and then select **OK**.



Review the selected data to ensure that you are satisfied with it, then select **Close & Apply** to load the data into Power BI Desktop.



The data now resembles a table with rows and columns. Data from Cosmos DB can now be related to data from other data sources and can eventually be used in a Power BI report.

Get data from applications

To support their daily operations, organizations frequently use a range of software applications, such as SharePoint, OneDrive, Dynamics 365, Google Analytics and so on. These applications produce their own data. Power BI can combine the data from multiple applications to produce more meaningful insights and reports.

Scenario:

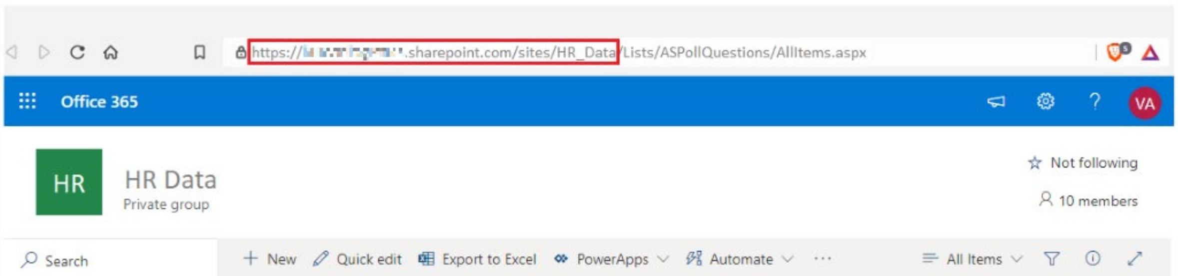
Tailwind Traders uses SharePoint to collaborate and store sales data. It's the start of the new financial year and the sales managers want to enter new goals for the sales team. The form that the leadership uses exists in SharePoint. You are required to establish a connection to this data within Power BI Desktop, so that the sales goals can be used alongside other sales data to determine the health of the sales pipeline.

The following sections examine how to use the Power BI Desktop **Get Data** feature to connect to data sources that are produced by external applications. To illustrate this process, an example is provided that shows how to connect to a SharePoint site and import data from an online list.

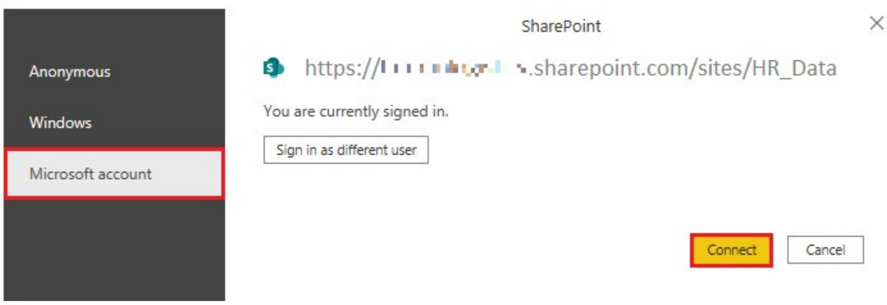
Connect to data in an application

When connecting to data in an application, you would begin in the same way as you would when connecting to the other data sources: by selecting the **Get data** feature in Power BI Desktop. Then, select the option that you need from the **Online Services** category. In this example, you select **SharePoint Online List**.

After you have selected **Connect**, you'll be asked for your SharePoint URL. This URL is the one that you use to sign into your SharePoint site through a web browser. You can copy the URL from your SharePoint site and paste it into the connection window in Power BI. You do not need to enter your full URL file path; you only need to load your site URL because, when you are connected, you can select the specific list that you want to load. Depending on the URL that you copied, you might need to delete the last part of your URL, as illustrated in the following image.



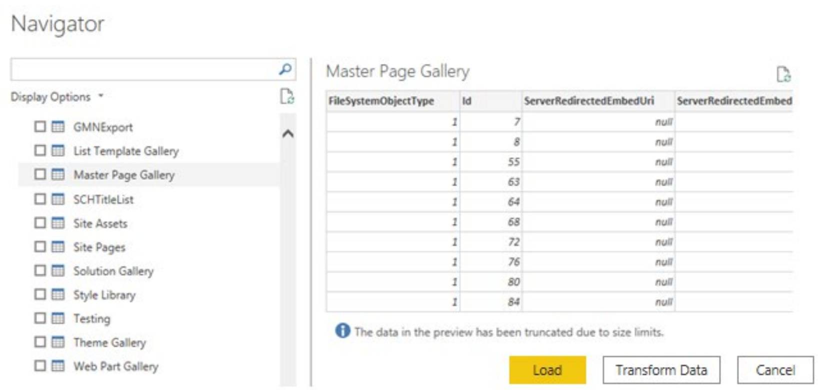
After you have entered your URL, select **OK**. Power BI needs to authorize the connection to SharePoint, so sign in with your Microsoft account and then select **Connect**.

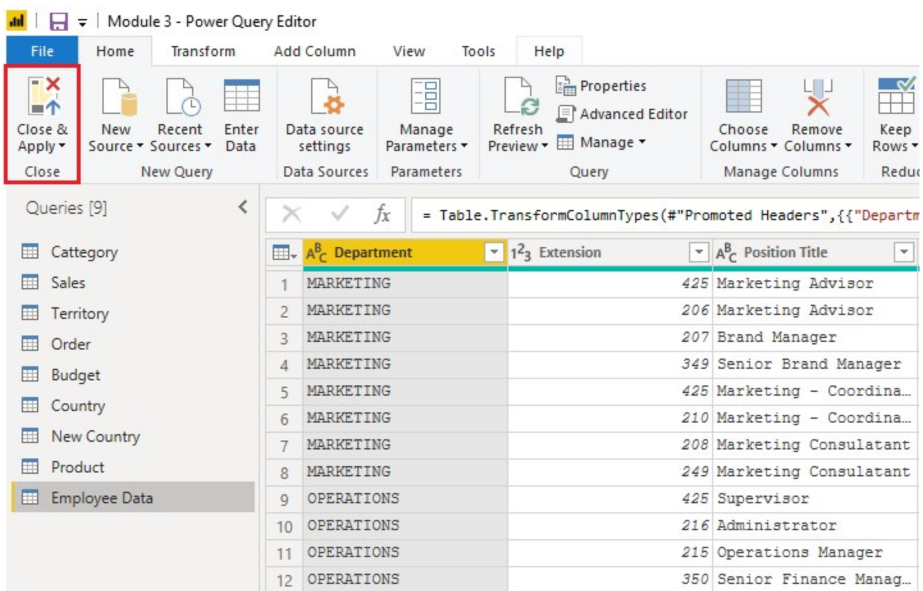


Choose the application data to import

After Power BI has made the connection with SharePoint, the **Navigator** window appears, as it does when you connect to other data sources. The window displays the tables and entities within your SharePoint site. Select the list that you want to load into Power BI Desktop. Similar to when you import from other data sources, you have the option to automatically load your data into Power BI model or launch the Power Query Editor to transform your data before loading it.

For this example, you select the **Load** option.





When you are satisfied with your data, select the **Close & Apply** button to apply your changes and load your data into Power BI Desktop.

Get data from Analysis Services

Azure Analysis Services is an Azure product that allows you to ingest data from multiple data sources, build relationships between the data, and creates calculations on the data. The calculations are built using data analysis expressions (DAX). Azure Analysis Services is similar to the data modeling and storage technology in Power BI.

To resume the scenario, Tailwind Traders uses Azure Analysis Services to store financial projection data. You've been asked to compare this data with actual sales data in a different database. Getting data from Azure Analysis Services cubes is similar to getting data from SQL Server, in that you can:

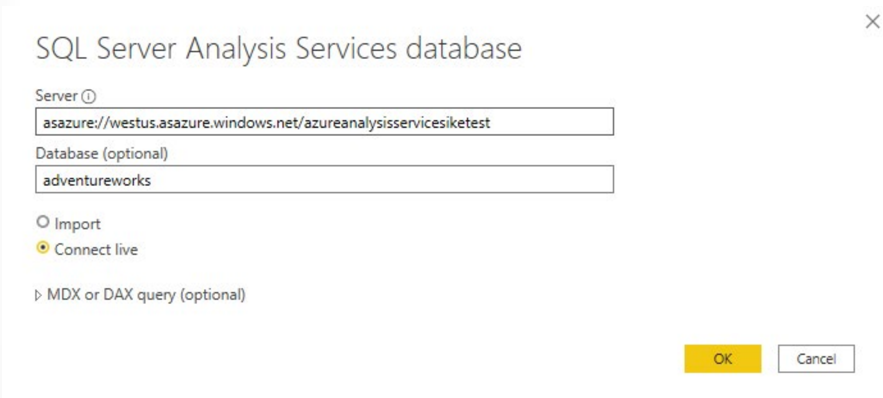
- Authenticate to the server.
- Pick the cube you want to use.
- Select which tables you need.

Notable differences between Azure Analysis Services cubes and SQL Server are:

- Analysis Services cubes have calculations already in the cube, which will be discussed in more detail later.
- If you don't need an entire table, you can query the data directly. Instead of using Transact-SQL (T-SQL) to query the data, like you would in SQL Server, you can use multi-dimensional expressions (MDX) or data analysis expressions (DAX).
- You don't need to use the **Get Data** button in Power BI Desktop.

Connect to data in Azure Analysis Services

As previously mentioned, you use the **Get data** feature in Power BI Desktop. When you select **Analysis Services**, you are prompted for the server address and the database name with two options: **Import** and **Connect live**.



Connect live is a new option in Azure Analysis Services. Azure Analysis Services uses the tabular model and DAX to build calculations, similar to Power BI. These models are compatible with one another. Using the Connect live option helps you keep the data and DAX calculations in their original location, without having to import them all into Power BI. Azure Analysis Services can have a fast refresh schedule , which means that when data is refreshed in the service, Power BI reports will immediately be updated, without the need to initiate a Power BI refresh schedule. This process can improve the timeliness of the data in your report.

Similar to a relational database, you can choose the tables that you want to use. If you want to directly query the Azure Analysis Services model, you can use DAX or MDX.

Because you want to get data to other data in your organization , you will likely import the data directly into Power BI. An acceptable alternative is to import all other data that you want (from Excel, SQL Server, and so on) into the Azure Analysis Services model and then use a live connection. Using this approach, the data modeling and DAX measures are all performed in one place, and it's a much simpler and easier way to maintain your solution.

For more information on connecting Power BI to Azure Analysis Services,
please refer to **Connect with Power BI documentation.**[1]

1 https://docs.microsoft.com/azure/analysis-services/analysis-services-connect-pbi

Get data from Microsoft Dataverse

Microsoft Dataverse is a cloud-based, low-code data service and app platform, which allows you to leverage the security and connectivity of Microsoft services. Dataverse connects easily to all aspects of Microsoft Power Platform so that you can fully control, automate, and strengthen your business. With standard tables and columns, as well as the ability to easily define relationships between your data, Dataverse was built for powerful, scalable solutions.

Getting data from Microsoft Dataverse

Connect to data in Microsoft Dataverse: Use the Get data feature in Power BI Desktop. When you select Power Platform, then Dataverse, select Connect to continue. You are prompted for the Environment domain with two options: Import and DirectQuery then select OK.
Select Sign in and then use your credentials when prompted. After you've successfully signed in, select Connect.
In the Navigator screen, enter the name of the table to locate it. Select the table by selecting the check box next to the name.
Select Transform Data to open the table in Power Query Editor.
Power Query Editor allows you to clean the data and helps make it easier for you to create charts, graphs, and other visualizations.

Because you want to get data to other data in your organization, you will likely import the data directly into Power BI. Using this approach, the data modeling and DAX measures are all performed in one place, and it's a much simpler and easier way to maintain your solution.

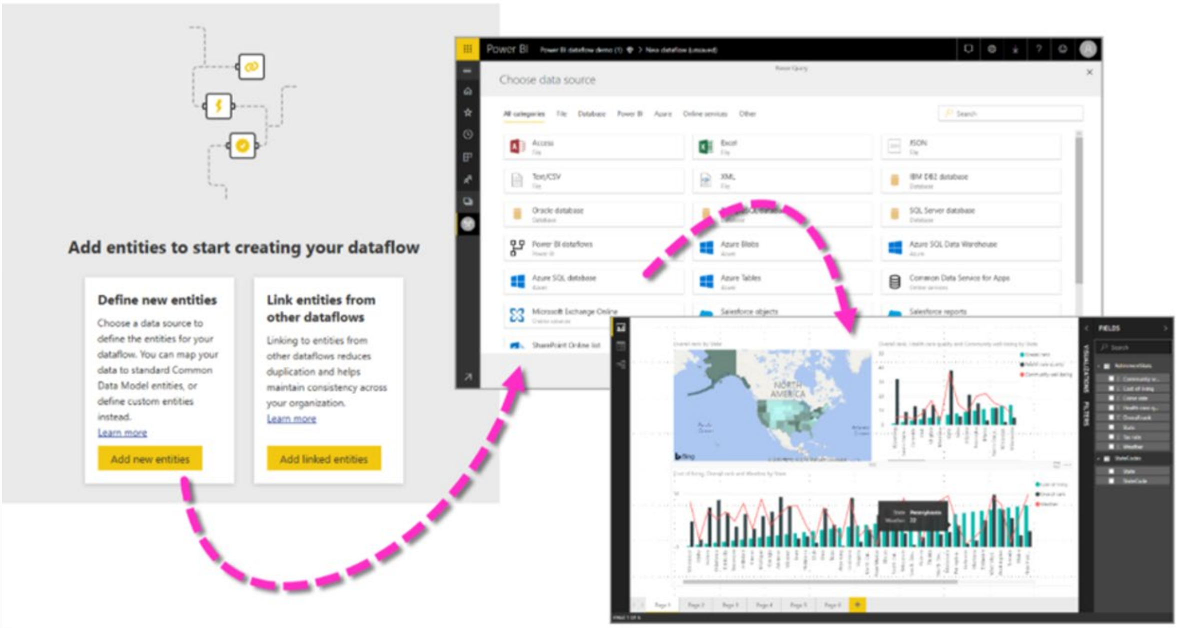
Get data from a dataflow

A dataflow is a collection of tables that are created and managed in workspaces in the Power BI service. A table is a set of columns that are used to store data, much like a table within a database. You can add and edit tables in your dataflow, and manage data refresh schedules, directly from the workspace in which your dataflow was created.

Getting data from dataflows

Connect to data in dataflows: Use the Get data feature in Power BI Desktop. When you select Power Platform, then dataflows, select Connect to continue.
Select Sign in and then use your credentials when prompted. After you've successfully signed in, select Connect.
In the Navigator screen, enter the name of the table to locate it. Select the table by selecting the check box next to the name.
Select Transform Data to open the table in Power Query Editor.

Power Query Editor allows you to clean the data and helps make it easier for you to create charts, graphs, and other visualizations.



Knowledge Check

Question 1

Which query language do you use to extract data from Microsoft SQL Server?

☐ DAX

☐ T-SQL

☐ MDX

Question 2

You're creating a Power BI report with data from an Azure Analysis Services Cube. When the data refreshes in the cube, you would like to see it immediately in the Power BI report. How should you connect?

☐ Connect Live

☐ Import

☐ Direct Query

Optimize Performance

Select a storage mode

The most popular way to use data in Power BI is to import it into a Power BI dataset. Importing the data means that the data is stored in the Power BI file and gets published along with the Power BI reports. This process helps make it easier for you to interact directly with your data. However, this approach might not work for all organizations.

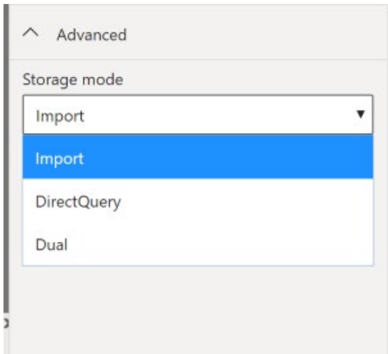
To continue with the scenario, you are building Power BI reports for the Sales department at Tailwind Traders, where importing the data is not an ideal method. The first task you need to accomplish is to create your datasets in Power BI so you can build visuals and other report elements. The Sales department has many different datasets of varying sizes. For security reasons, you are not allowed to import local copies of the data into your reports, so directly importing data is no longer an option. Therefore, you need to create a direct connection to the Sales department's data source. The following section describes how you can ensure that these business requirements are satisfied when you are importing data into Power BI.

However, sometimes there may be security requirements around your data that make it impossible to directly import a copy. Or your datasets may simply be too large and would take too long to load into Power BI, and you want to avoid creating a performance bottleneck. Power BI solves these problems by using the DirectQuery storage mode, which allows you to query the data in the data source directly and not import a copy into Power BI. DirectQuery is useful because it ensures you are always viewing the most recent version of the data.

The three different types of storage modes you can choose from:

- Import
- DirectQuery
- Dual (Composite)

You can access storage modes by switching to the **Model** view, selecting a data table, and in the resulting Properties pane, selecting which mode that you want to use from the **Storage mode** drop-down list, as shown in the following visual.



Let's take a closer look at the different types of Storage Modes.

Import mode

The Import mode allows you to create a local Power BI copy of your datasets from your data source. You can use all Power BI service features with this storage mode, including Q&A and Quick Insights. However, data refreshes must be done manually. Import mode is the default for creating new Power BI reports.

DirectQuery mode

The DirectQuery option is useful when you do not want to save local copies of your data because your data will not be cached. Instead, you can query the specific tables that you will need by using native Power BI queries, and the required data will be retrieved from the underlying data source. Essentially, you are creating a direct connection to the data source. Using this model ensures that you are always viewing the most up-to-date data, and that all security requirements are satisfied. Additionally, this mode is suited for when you have large datasets to pull data from. Instead of slowing down performance by having to load large amounts of data into Power BI, you can use DirectQuery to create a connection to the source, solving data latency issues as well.

Dual (Composite mode)

In Dual mode, you can identify some data to be directly imported and other data that must be queried. Any table that is brought in to your report is a product of both Import and DirectQuery modes. Using the Dual mode allows Power BI to choose the most efficient form of data retrieval.

For more information regarding Storage Modes, please refer to **Storage Modes**[2].

Fix performance issues

Occasionally, organizations will need to address performance issues when running reports. Power BI provides the Performance Analyzer tool to help fix problems and streamline the process.

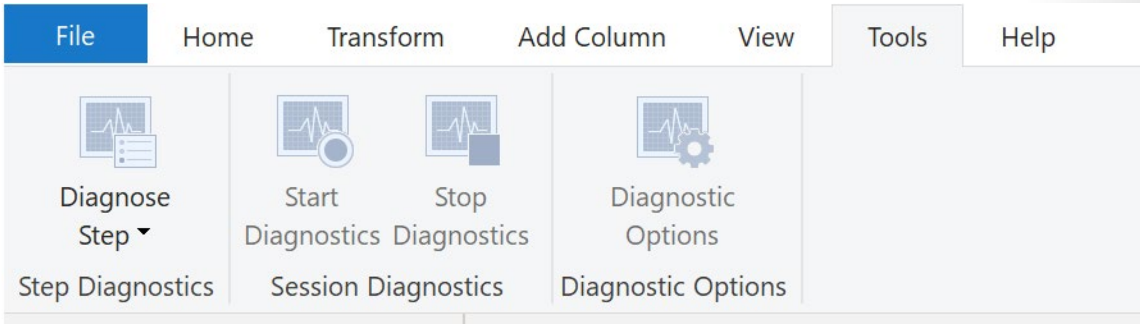
Consider the scenario where you are building reports for the Sales team in your organization. You've imported your data, which is in several tables within the Sales team's SQL database, by creating a data connection to the database through DirectQuery. When you create preliminary visuals and filters, you notice that some tables are queried faster than others, and some filters are taking longer to process compared to others.

Query diagnostics

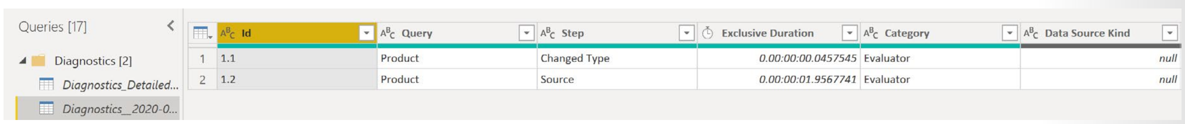
Another tool that you can use to study query performance is *query diagnostics*. This feature allows you to determine what bottlenecks (if any) exist while loading and transforming your data, refreshing your data in Power Query, running SQL statements in Query Editor, and so on.

To access query diagnostics in Power Query Editor, go to **Tools** in the Home ribbon. When you are ready to begin transforming your data or making other edits in Power Query Editor, select **Start Diagnostics** on the **Session Diagnostics** tab. When you are finished, make sure that you select **Stop Diagnostics**.

2 https://docs.microsoft.com/power-bi/transform-model/desktop-storage-mode



Selecting **Diagnose Step** shows you the length of time that it takes to run that step, as shown in the following image. This selection can tell you if a step takes longer to complete than others, which then serves as a starting point for further investigation.



This tool is useful when you want to analyze performance on the Power Query side for tasks such as loading datasets, running data refreshes, or running other transformative tasks.

Other techniques to optimize performance

Other ways to optimize query performance in Power BI include:

- **Process as much data as possible in the original data source**. Power Query and Power Query Editor allow you to process the data; however, the processing power that is required to complete this task might lower performance in other areas of your reports. Generally, a good practice is to process, as much as possible, in the native data source.
- **Use native SQL queries**. When using DirectQuery for SQL databases, such as the case for our scenario, make sure that you are not pulling data from stored procedures or common table expressions (CTEs).
- **Separate date and time, if bound together**. If any of your tables have columns that combine date and time, make sure that you separate them into distinct columns before importing them into Power BI. This approach will increase compression abilities.

For more information, refer to **Query Folding Guidance**[3] and **Query Folding**[4].

Optimize performance in Power Query

The performance in Power Query depends on the performance at the data source level. The variety of data sources that Power Query offers is very wide, and the performance tuning techniques for each source are equally wide. For instance, if you extract data from a Microsoft SQL Server, you should follow the performance tuning guidelines for the product. Good SQL Server performance tuning techniques includes index creation, hardware upgrades, execution plan tuning, and data compression. These topics

3 https://docs.microsoft.com/power-bi/guidance/power-query-folding
4 https://docs.microsoft.com/power-query/power-query-folding

are beyond the scope here, and are covered only as an example to build familiarity with your data source and reap the benefits when using Power BI and Power Query.

Power Query takes advantage of good performance at the data source through a technique called Query Folding.

Query folding

The query folding within Power Query Editor helps you increase the performance of your Power BI reports. *Query folding* is the process by which the transformations and edits that you make in Power Query Editor are simultaneously tracked as native queries, or simple **Select** SQL statements, while you are actively making transformations. The reason for implementing this process is to ensure that these transformations can take place in the original data source server and do not overwhelm Power BI computing resources.

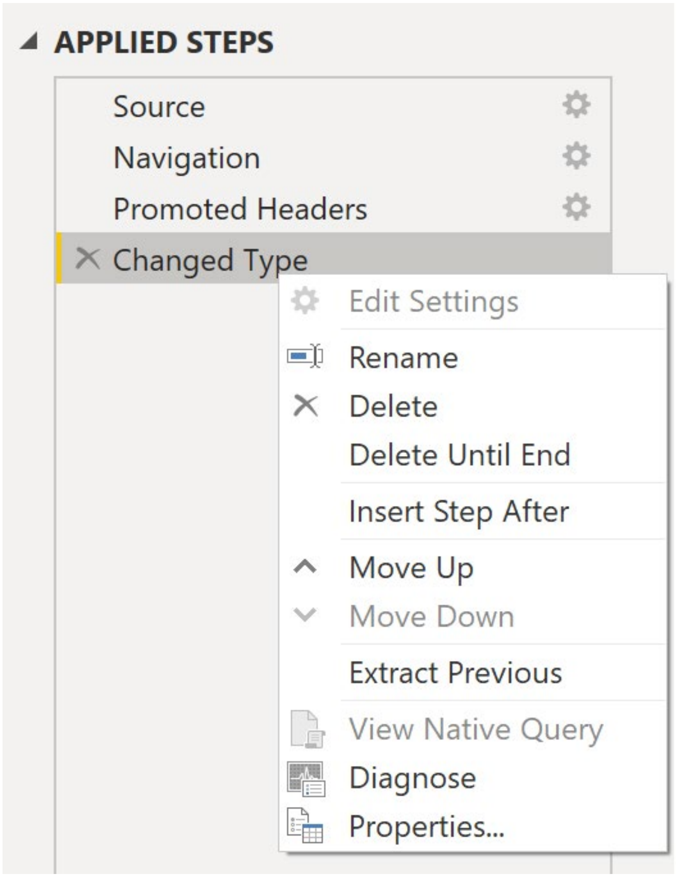
You can use Power Query to load data into Power BI. Using Power Query Editor you can then make further transformations to your data, such as renaming or deleting columns, appending, parsing, filtering, or grouping your data.

Consider a scenario where you've renamed a few columns in the Sales data and merged a city and state column together in the “city state” format. Meanwhile, the query folding feature tracks those changes in native queries. Then, when you load your data, the transformations take place independently in the original source, this ensures that performance is optimized in Power BI.

The benefits to query folding include:

- **More efficiency in data refreshes and incremental refreshes**. When you import data tables by using query folding, Power BI is better able to allocate resources and refresh the data faster because Power BI does not have to run through each transformation locally.
- **Automatic compatibility with DirectQuery and Dual storage modes**. All DirectQuery and Dual storage mode data sources must have the back-end server processing abilities to create a direct connection, which means that query folding is an automatic capability that you can use. If all transformations can be reduced to a single **Select** statement, then query folding can occur.

The following scenario shows query folding in action. In this scenario, you apply a set of queries to multiple tables. After you add a new data source by using Power Query, and you are directed to the Power Query Editor, you go to the **Query Settings** pane and right-click the last applied step, as shown in the following figure.



If the **View Native Query** option is not available (not displayed in bold type), that query folding is not possible for this step, and you will have to work backward in the **Applied Steps** area until you reach the step in which **View Native Query** is available (displays in bold type). This process will reveal the native query that is used to transform the dataset.

Native queries are not possible for the following transformations:

- Adding an index column
- Merging and appending columns of different tables with two different sources
- Changing the data type of a column
- Running complex DAX functions

A good guideline to remember is that if you can translate a transformation into a **Select** SQL statement, which includes operators and clauses such as GROUP BY, SORT BY, WHERE, UNION ALL, and JOIN, you can use query folding.

While query folding is one option to optimize performance when retrieving, importing, and preparing data, another option is query diagnostics discussed earlier.

Knowledge Check

Question 1

Which storage mode leaves the data at the data source?

☐ Import

☐ Direct Query

☐ Dual

Question 2

Which technology improves performance by generating a single query statement to retrieve and transform source data?

☐ Query folding

☐ Adding index columns

☐ Adding custom columns with complex logic

Question 3

What can you do to improve performance when you are getting data in Power BI?

☐ Only pull data into the Power BI service, not Power BI Desktop

☐ Use the Select SQL statement in your SQL queries when you are pulling data from a relational database

☐ Combine date and time columns into a single column

☐ Do some calculations in the original data source

Resolve Data Errors

Identify and resolve data import errors

While importing data into Power BI, you may encounter errors resulting from factors such as:

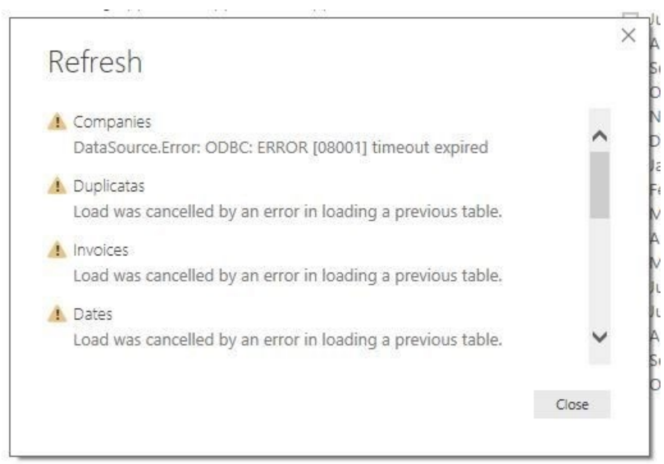
- Power BI imports from numerous data sources.
- Each data source might have dozens (and sometimes hundreds) of different error messages.
- Other components can cause errors, such as hard drives, networks, software services, and operating systems.
- Data can often not comply with any specific schema.

The following sections cover some of the more common error messages that you might encounter in Power BI.

Query timeout expired

Relational source systems often have many people who are concurrently using the same data in the same database. Some relational systems and their administrators seek to limit a user from monopolizing all hardware resources by setting a query timeout. These timeouts can be configured for any timespan, from as little as five seconds to as much as 30 minutes or more.

For instance, if you're pulling data from your organization's SQL Server, you might see the error shown in the following figure.



Power BI Query Error: Timeout expired

This error indicates that you've pulled too much data according to your organization's policies. Administrators incorporate this policy to avoid slowing down a different application or suite of applications that might also be using that database.

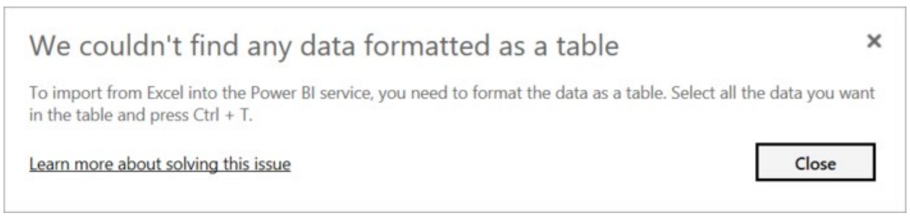
You can resolve this error by pulling fewer columns or rows from a single table. While you are writing SQL statements, it might be a common practice to include groupings and aggregations. You can also join multiple tables in a single SQL statement. Additionally, you can perform complicated subqueries and nested queries in a single statement. These complexities add to the query processing requirements of the relational system and can greatly elongate the time of implementation.

If you need the rows, columns, and complexity, consider taking small chunks of data and then bringing them back together by using Power Query. For instance, you can combine half the columns in one query and the other half in a different query. Power Query can merge those two queries back together after you are finished.

We couldn't find any data formatted as a table

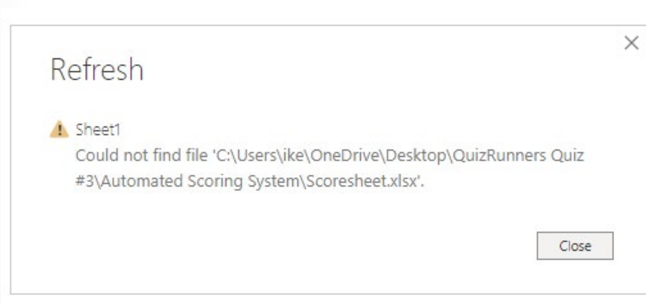
Occasionally, you may encounter the “We couldn't find any data formatted as a table” error while importing data from Microsoft Excel. Fortunately, this error is self-explanatory. Power BI expects to find data formatted as a table from Excel. The error event tells you the resolution. Perform the following steps to resolve the issue:

1. Open your Excel workbook, and highlight the data that you want to import.
2. Press the **Ctrl-T** keyboard shortcut. The first row will likely be your column headers.
3. Verify that the column headers reflect how you want to name your columns. Then, try to import data from Excel again. This time, it should work.



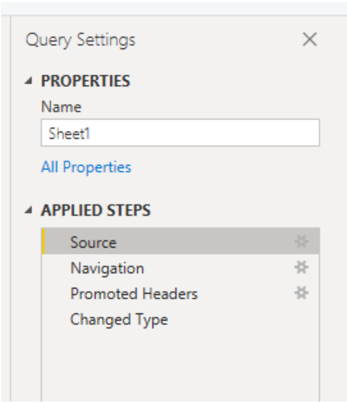
Could not find file

While importing data from a file, you may get the “Could not find file” error.

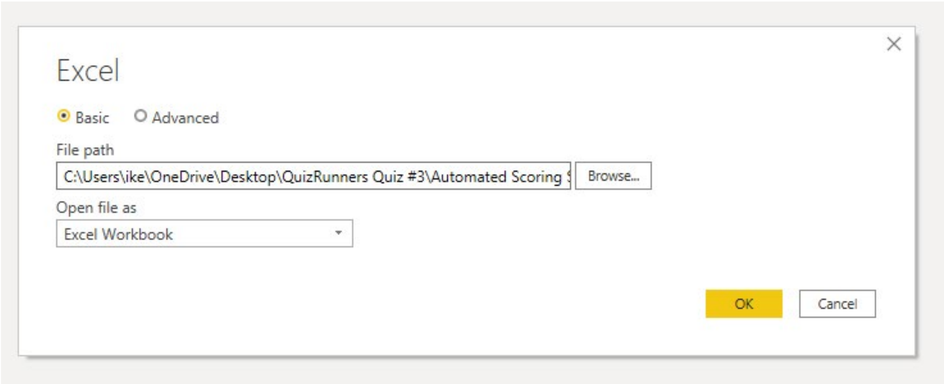


Usually, this error is caused by the file moving locations or the permissions to the file changing. If the cause is the former, you need to find the file and change the source settings.

1. Open Power Query by selecting the **Transform Data** button in Power BI.
2. Highlight the query that is creating the error.
3. On the left, under **Query Settings**, select the gear icon next to **Source**.



4. Change the file location to the new location.



Data type errors

Sometimes, when you import data into Power BI, the columns appear blank. This situation happens because of an error in interpreting the data type in Power BI. The resolution to this error is unique to the data source. For instance, if you are importing data from SQL Server and see blank columns, you could try to convert to the correct data type in the query.

Instead of using this query:

```
SELECT CustomerPostalCode FROM Sales.Customers
```

Use this query:

```
SELECT CAST(CustomerPostalCode as varchar(10)) FROM Sales.Customers
```

By specifying the correct type at the data source, you eliminate many of these common data source errors.

You may encounter different types of errors in Power BI that are caused by the diverse data source systems where your data resides. If you encounter an error that was not discussed in the preceding sections, you can search Microsoft documentation for the error message to find the resolution you need. he preceding sections, you can search Microsoft documentation for the error message to find the resolution you need.

Knowledge Check

Question 1

What type of import error might leave a column blank?

- ☐ Keep errors
- ☐ Unpivot columns
- ☐ Data type error

Answers

Question 1

Which query language do you use to extract data from Microsoft SQL Server?

- ☐ DAX
- ■ T-SQL
- ☐ MDX

Question 2

You're creating a Power BI report with data from an Azure Analysis Services Cube. When the data refreshes in the cube, you would like to see it immediately in the Power BI report. How should you connect?

- ■ Connect Live
- ☐ Import
- ☐ Direct Query

Question 1

Which storage mode leaves the data at the data source?

- ☐ Import
- ■ Direct Query
- ☐ Dual

Question 2

Which technology improves performance by generating a single query statement to retrieve and transform source data?

- ■ Query folding
- ☐ Adding index columns
- ☐ Adding custom columns with complex logic

Question 3

What can you do to improve performance when you are getting data in Power BI?

- ☐ Only pull data into the Power BI service, not Power BI Desktop
- ☐ Use the Select SQL statement in your SQL queries when you are pulling data from a relational database
- ☐ Combine date and time columns into a single column
- ■ Do some calculations in the original data source

Question 1

What type of import error might leave a column blank?

- ☐ Keep errors
- ☐ Unpivot columns
- ■ Data type error

Module 3 Clean, Transform, and Load Data in Power BI

Data Shaping

Introduction

Consider the scenario where you have imported data into Power BI from several different sources and, when you examine the data, it is not prepared for analysis. What could make the data unprepared for analysis?

When examining the data, you discover several issues, including:

- A column called **Employment status** only contains numerals.
- Several columns contain errors.
- Some columns contain null values.
- The customer ID in some columns appears as if it was duplicated repeatedly.
- A single address column has combined street address, city, state, and zip code.

You start working with the data, but every time you create visuals on reports, you get bad data, incorrect results, and simple reports about sales totals are wrong.

Dirty data can be overwhelming and, though you might feel frustrated, you decide to get to work and figure out how to make this data model as pristine as possible.

Fortunately, Power BI and Power Query offer you a powerful environment to clean and prepare the data. Clean data has the following advantages:

- Measures and columns produce more accurate results when they perform aggregations and calculations.

- Tables are organized, where users can find the data in an intuitive manner.
- Duplicates are removed, making data navigation simpler. It will also produce columns that can be used in slicers and filters.
- A complicated column can be split into two, simpler columns. Multiple columns can be combined into one column for readability.
- Codes and integers can be replaced with human readable values.

In this module, you will learn how to:

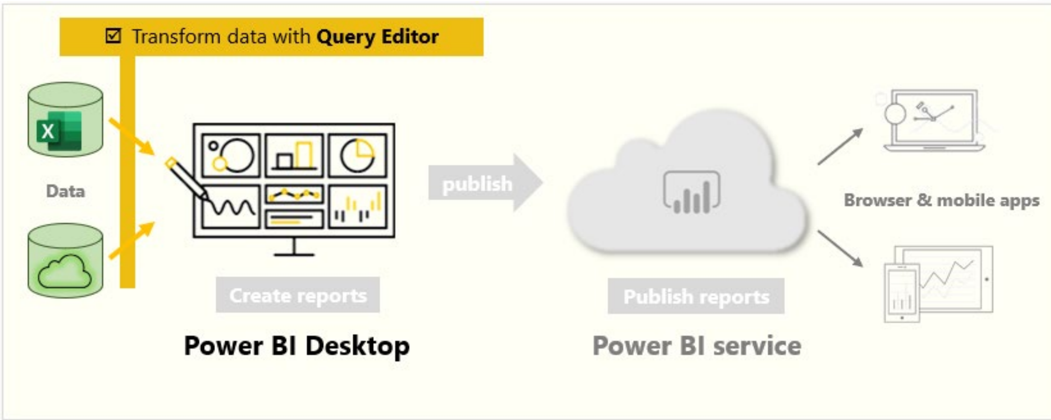
- Resolve inconsistencies, unexpected or null values, and data quality issues.
- Profile data so you can learn more about a specific column before using it.
- Evaluate and transform column data types.
- Apply data shape transformations to table structures.
- Combine queries.
- Apply user-friendly naming conventions to columns and queries.

Identify column header and names

Power Query Editor in Power BI Desktop allows you to shape (transform) your imported data. You can accomplish actions such as renaming columns or tables, changing text to numbers, removing rows, setting the first row as headers, and much more. It is important to shape your data to ensure that it meets your needs and is suitable for use in reports.

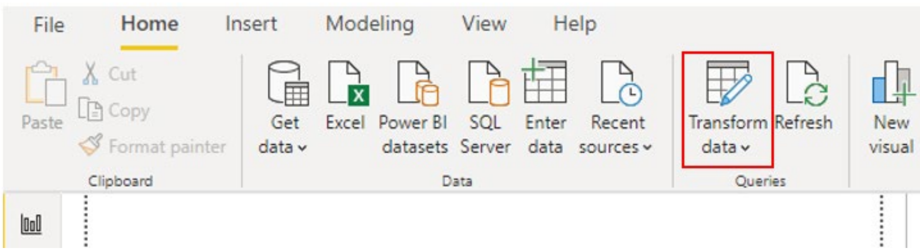
You have loaded raw sales data from two sources into a Power BI model. Some of the data came from a .csv file that was created manually in Microsoft Excel by the Sales team. The other data was loaded through a connection to your organization's Enterprise Resource Planning (ERP) system. Now, when you look at the data in Power BI Desktop, you notice that it's in disarray; some data that you don't need and some data that you do need are in the wrong format.

You need to use Power Query Editor to clean up and shape this data before you can start building reports.



Get started with Power Query Editor

To start shaping your data, open Power Query Editor by selecting the **Transform data** option on the **Home** tab of Power BI Desktop.



In Power Query Editor, the data in your selected query displays in the middle of the screen and, on the left side, the **Queries** pane lists the available queries (tables).

When you work in Power Query Editor, all steps that you take to shape your data are recorded. Then, each time the query connects to the data source, it automatically applies your steps, so your data is always shaped the way that you specified. Power Query Editor only makes changes to a particular view of your data, so you can feel confident about changes that are being made to your original data source. You can see a list of your steps on the right side of the screen, in the **Query Settings** pane, along with the query's properties.

The Power Query Editor ribbon contains many buttons you can use to select, view, and shape your data.

To learn more about the available features and functions, see **The query ribbon**[1].

NOTE: In Power Query Editor, the right-click context menus and **Transform** tab in the ribbon provide many of the same options.

1 https://docs.microsoft.com/power-query/power-query-quickstart-using-power-bi#the-query-ribbon

Identify column headers and names

The first step in shaping your initial data is to identify the column headers and names within the data and then evaluate where they are located to ensure that they are in the right place.

In the following screenshot, the source data in the SalesTarget.csv file had a target categorized by products and a subcategory split by months, both of which are organized into columns.

	Column1	Column2	Column3	Column4	Column5	Column6	Column7
1			January	February	March	April	May
2							
3	ProductSubcategoryID	Name					
4	1	Mountain Bikes	780000	790000	800000	810000	820000
5	2	Road Bikes	4500	5000	5500	6000	6500
6	3	Touring Bikes	501000	502000	503000	504000	505000
7	4	Handlebars	1100	1200	1300	1400	1500
8	5	Bottom Brackets	1100	1200	1300	1400	1500
9	6	Brakes	1100	1200	1300	1400	1500
10	7	Chains	1100	1200	1300	1400	1500
11	8	Cranksets	1100	1200	1300	1400	1500
12	9	Derailleurs	1100	1200	1300	1400	1500

However, you notice that the data did not import as expected.

A	B	C	D	E	F	G	H	I	J	K	L	M	N
ProductSubcategoryID	Name	January	February	March	April	May	June	July	August	September	October	November	December
1	Mountain Bikes	780000	790000	800000	810000	820000	830000	840000	850000	860000	870000	880000	890000
2	Road Bikes	4500	5000	5500	6000	6500	7000	7500	8000	8500	9000	9500	10000
3	Touring Bikes	501000	502000	503000	504000	505000	506000	507000	508000	509000	510000	511000	512000
4	Handlebars	1100	1200	1300	1400	1500	1600	1700	1800	1900	2000	2100	2200
5	Bottom Brackets	1100	1200	1300	1400	1500	1600	1700	1800	1900	2000	2100	2200
6	Brakes	1100	1200	1300	1400	1500	1600	1700	1800	1900	2000	2100	2200
7	Chains	1100	1200	1300	1400	1500	1600	1700	1800	1900	2000	2100	2200
8	Cranksets	1100	1200	1300	1400	1500	1600	1700	1800	1900	2000	2100	2200
9	Derailleurs	1100	1200	1300	1400	1500	1600	1700	1800	1900	2000	2100	2200

Consequently, the data is difficult to read. A problem has occurred with the data in its current state because column headers are in different rows (marked in red), and several columns have undescriptive names, such as **Column1**, **Column2**, and so on.

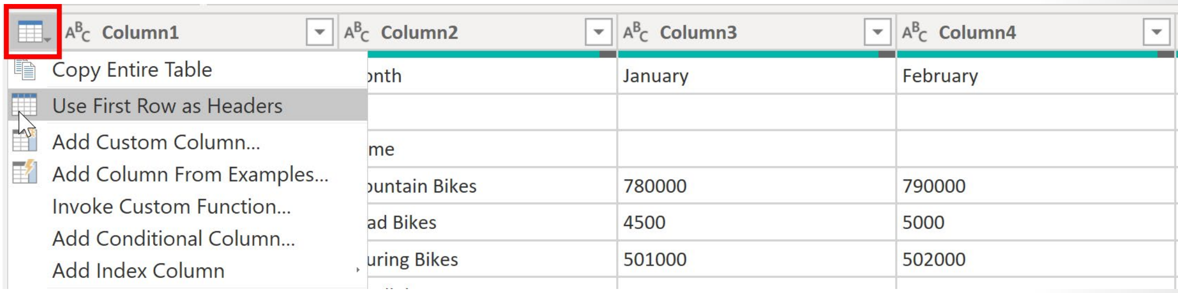
When you have identified where the column headers and names are located, you can make changes to reorganize the data.

Shaping table structure

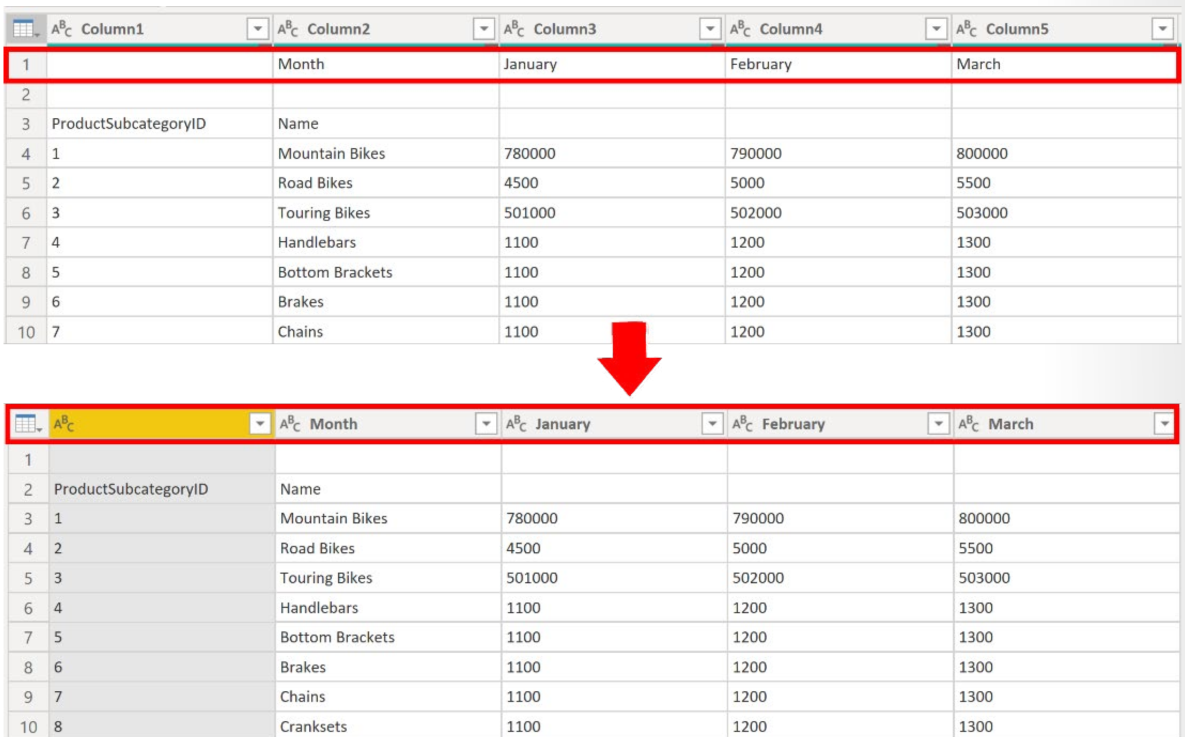
Promote headers

When a table is created in Power BI Desktop, Power Query Editor assumes that all data belongs in table rows. However, a data source might have a first row that contains column names, which is what happened in the previous SalesTarget example. To correct this inaccuracy, you need to promote the first table row into column headers.

You can promote headers in two ways: by selecting the **Use First Row as Headers** option on the **Home** tab or by selecting the drop-down button next to **Column1** and then selecting **Use First Row as Headers**.



The following image illustrates how the **Use First Row as Headers** feature impacts the data:



	Column1	Column2	Column3	Column4	Column5
1		Month	January	February	March
2					
3	ProductSubcategoryID	Name			
4	1	Mountain Bikes	780000	790000	800000
5	2	Road Bikes	4500	5000	5500
6	3	Touring Bikes	501000	502000	503000
7	4	Handlebars	1100	1200	1300
8	5	Bottom Brackets	1100	1200	1300
9	6	Brakes	1100	1200	1300
10	7	Chains	1100	1200	1300

		Month	January	February	March
1					
2	ProductSubcategoryID	Name			
3	1	Mountain Bikes	780000	790000	800000
4	2	Road Bikes	4500	5000	5500
5	3	Touring Bikes	501000	502000	503000
6	4	Handlebars	1100	1200	1300
7	5	Bottom Brackets	1100	1200	1300
8	6	Brakes	1100	1200	1300
9	7	Chains	1100	1200	1300
10	8	Cranksets	1100	1200	1300

Rename columns

The next step in shaping your data is to examine the column headers. You might discover that one or more columns have the wrong headers, a header has a spelling error, or the header naming convention is not consistent or user-friendly.

Refer to the previous screenshot, which shows the impact of the **Use First Row as Headers** feature. Notice that the column that contains the subcategory **Name** data now has **Month** as its column header. This column header is incorrect, so it needs to be renamed.

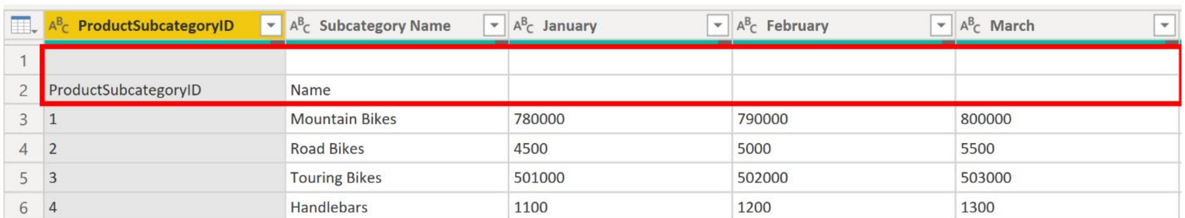
You can rename column headers in two ways. One approach is to right-click the header, select **Rename**, edit the name, and then press **Enter**. Alternatively, you can double-click the column header and overwrite the name withthe correct name.

You can also work around this issue by removing (skipping) the first two rows and then renaming the columns to the correct name.

Remove top rows

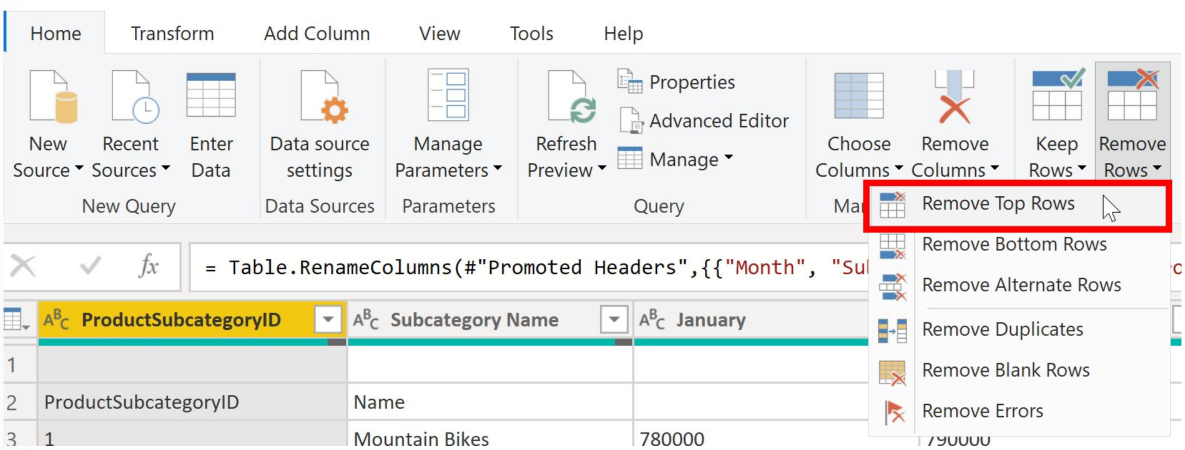
When shaping your data, you might need to remove some of the top rows, for example, if they are blank or if they contain data that you do not need in your reports.

Continuing with the SalesTarget example, notice that the first row is blank (it has no data) and the second row has data that is no longer required.



	ProductSubcategoryID	Subcategory Name	January	February	March
1					
2	ProductSubcategoryID	Name			
3	1	Mountain Bikes	780000	790000	800000
4	2	Road Bikes	4500	5000	5500
5	3	Touring Bikes	501000	502000	503000
6	4	Handlebars	1100	1200	1300

To remove these excess rows, select **Remove Rows** > **Remove Top Rows** on the **Home** tab.



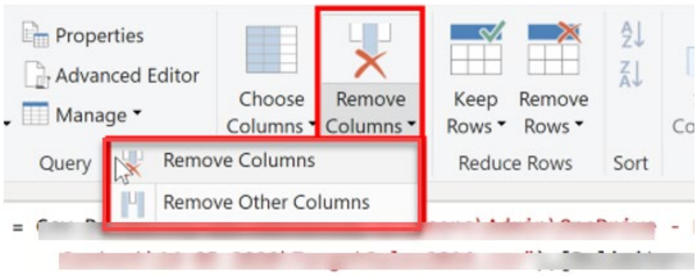
Remove columns

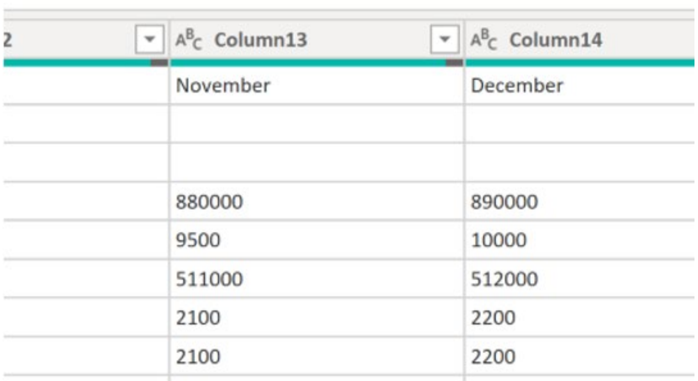
A key step in the data shaping process is to remove unnecessary columns. It is much better to remove columns as early as possible. One way to remove columns would be to limit the column when you get data from data source. For instance, if you are extracting data from a relational database by using SQL, you would want to limit the column that you extract by using a column list in the SELECT statement.

Removing columns at an early stage in the process rather than later is best, especially when you have established relationships between your tables. Removing unnecessary columns will help you to focus on the data that you need and help improve the overall performance of your Power BI Desktop datasets and reports.

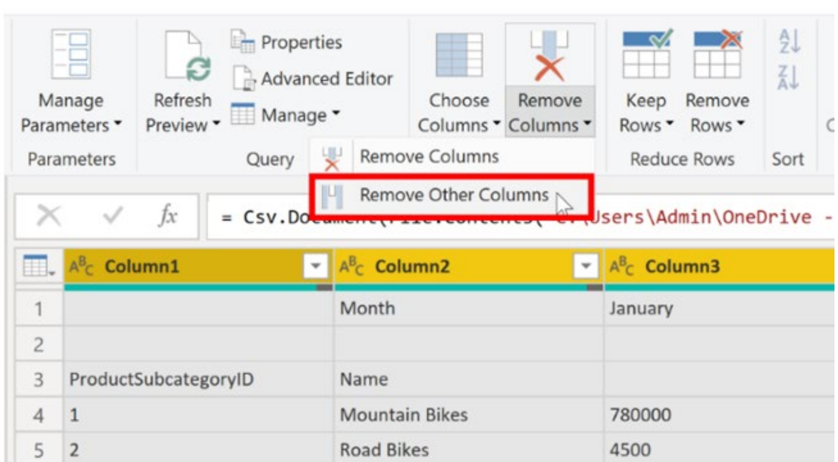
Examine each column and ask yourself if you really need the data that it contains. If you don't plan on using that data in a report, the column adds no value to your data model. Therefore, the column should be removed. You can always add the column later, if your requirements change over time.

You can remove columns in two ways. The first method is to select the columns that you want to remove and then, on the **Home** tab, select **Remove Columns**.





Alternatively, you can select the columns that you want to keep and then, on the **Home** tab, select **Remove Columns** > **Remove Other Columns**.



Pivot and Unpivot

Unpivot columns

Unpivoting is a useful feature of Power BI. You can use this feature with data from any data source, but you would most often use it when importing data from Excel. The following example shows a sample Excel document with sales data.

Year		2018		2019
January	$	15,370	$	16,063
February	$	15,950	$	12,161
March	$	13,862	$	14,180
April	$	18,530	$	6,516
May	$	5,203	$	19,395
June	$	5,928	$	19,324
July	$	14,736	$	15,939
August	$	6,243	$	15,390
Septemb	$	15,178	$	17,832
October	$	18,148	$	5,185
Novemb	$	8,014	$	9,299
Decembe	$	19,470	$	14,082

Though the data might initially make sense, it would be difficult to create a total of all sales combined from 2018 and 2019. Your goal would then be to use this data in Power BI with three columns: **Month**, **Year**, and **SalesAmount**.

When you import the data into Power Query, it will look like the following image.

	Year	2018	2019
1	January	15370	16063
2	February	15950	12161
3	March	13862	14180
4	April	18530	6516
5	May	5203	19395
6	June	5928	19324
7	July	14736	15939
8	August	6243	15390
9	September	15178	17832
10	October	18148	5185
11	November	8014	9299
12	December	19470	14082

Next, rename the first column to **Month**. This column was mislabeled because that header in Excel was labeling the 2018 and 2019 columns. Highlight the 2018 and 2019 columns, select the **Transform** tab in Power Query, and then select **Unpivot**.

Year	Attribute	Value
January	2018	15370
January	2019	16063
February	2018	15950
February	2019	12161
March	2018	13862
March	2019	14180
April	2018	18530
April	2019	6516
May	2018	5203
May	2019	19395
June	2018	5928
June	2019	19324
July	2018	14736
July	2019	15939
August	2018	6243
August	2019	15390
September	2018	15178
September	2019	17832
October	2018	18148
October	2019	5185
November	2018	8014
November	2019	9299
December	2018	19470
December	2019	14082

You can rename the **Attribute** column to **Year** and the **Value** column to **SalesAmount**.

Unpivoting streamlines the process of creating DAX measures on the data later. By completing this process, you have now created a simpler way of slicing the data with the **Year** and **Month** columns.

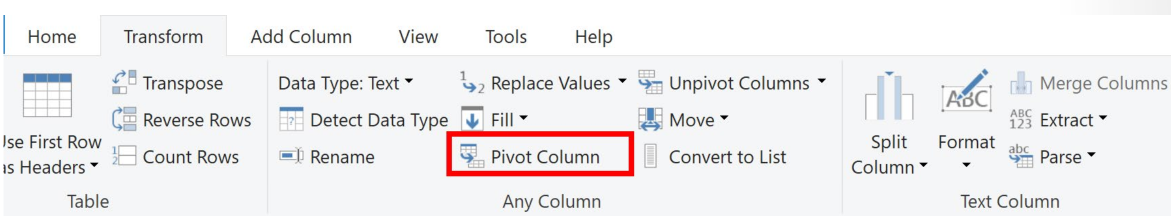
Pivot columns

If the data that you are shaping is flat (in other words, it has lot of detail but is not organized or grouped in any way), the lack of structure can complicate your ability to identify patterns in the data.

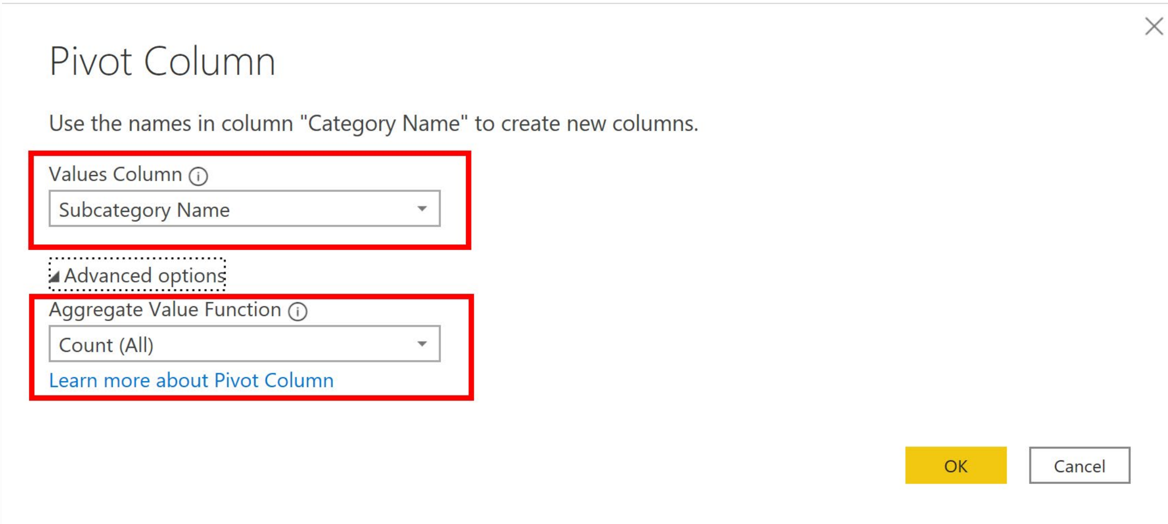
You can use the **Pivot Column** feature to convert your flat data into a table that contains an aggregate value for each unique value in a column. For example, you might want to use this feature to summarize data by using different math functions such as **Count**, **Minimum**, **Maximum**, **Median**, **Average**, or **Sum**.

In the SalesTarget example, you can pivot the columns to get the quantity of product subcategories in each product category.

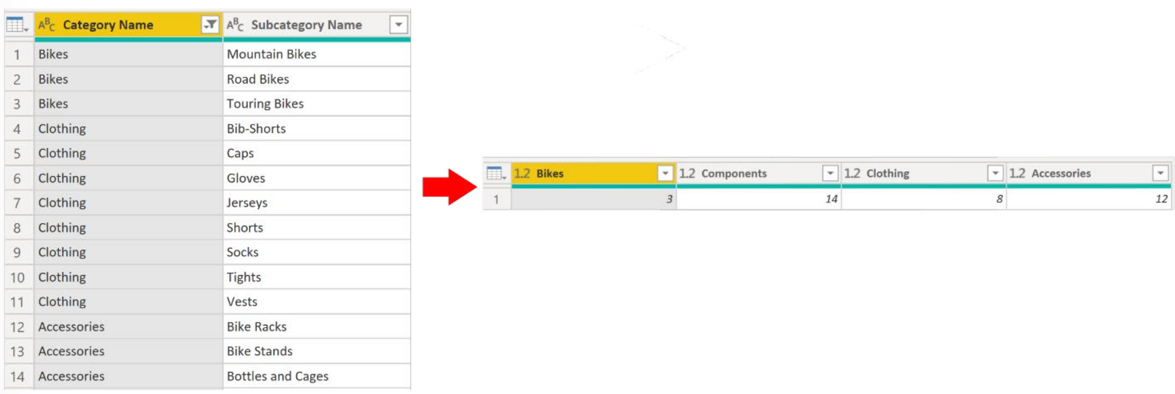
On the **Transform** tab, select **Transform > Pivot Columns**.



On the **Pivot Column** window that displays, select a column from the **Values Column** list, such as **Subcategory name**. Expand the advanced options and select an option from the **Aggregate Value Function** list, such as **Count (All)**, and then select **OK**.



The following image illustrates how the **Pivot Column** feature changes the way that the data is organized.



Power Query Editor records all steps that you take to shape your data, and the list of steps are shown in the **Query Settings** pane. If you have made all the required changes, select **Close & Apply** to close Power Query Editor and apply your changes to your data model. However, before you select **Close & Apply**, you can take further steps to clean up and transform your data in Power Query Editor. These additional steps are covered later in this module.

Knowledge Check

Question 1

The primary data preparation tool in Power BI is called what?

☐ Report editor

☐ Power Query editor

☐ Data editor

Question 2

The process of shaping data by converting your flat data into a table that contains an aggregation value for each unique value in a column is called what?

☐ Group by columns

☐ Pivot (pivoting a column)

☐ Manage aggregations

Question 3

What can be achieved by removing unnecessary rows and columns?

☐ It is not necessary to delete unnecessary rows and columns and it is a good practice to keep all metadata intact.

☐ Deleting unnecessary rows and columns can damage the structure of the data model.

☐ Deleting unnecessary rows and columns will reduce the dataset size and it is a good practice to load only necessary data into your data model.

Data Profiling

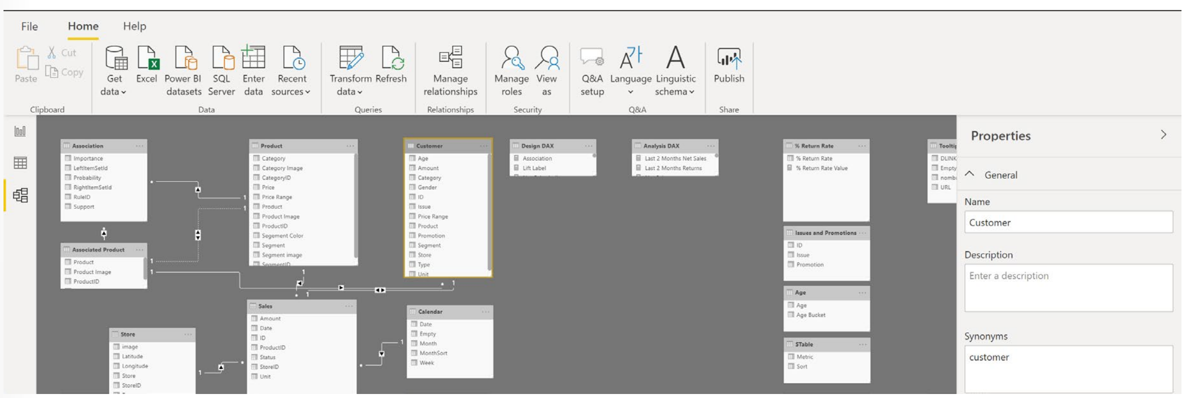
Profiling data

Profiling data is about studying the nuances of the data: determining anomalies, examining and developing the underlying data structures, and querying data statistics such as row counts, value distributions, minimum and maximum values, averages, and so on. This concept is important because it allows you to shape and organize the data so that interacting with the data and identifying the distribution of the data is uncomplicated, therefore helping to make your task of working with the data on the front end to develop report elements near effortless.

Assume that you are developing reports for the Sales team at your organization. You are uncertain how the data is structured and contained within the tables, so you want to profile the data behind the scenes before you begin developing the visuals. Power BI has inherent functionality that makes these tasks user-friendly and straightforward.

Examine data structures

Before you begin examining the data in Power Query Editor, you should first learn about the underlying data structures that data is organized in. You can view the current data model under the **Model** tab on Power BI Desktop.



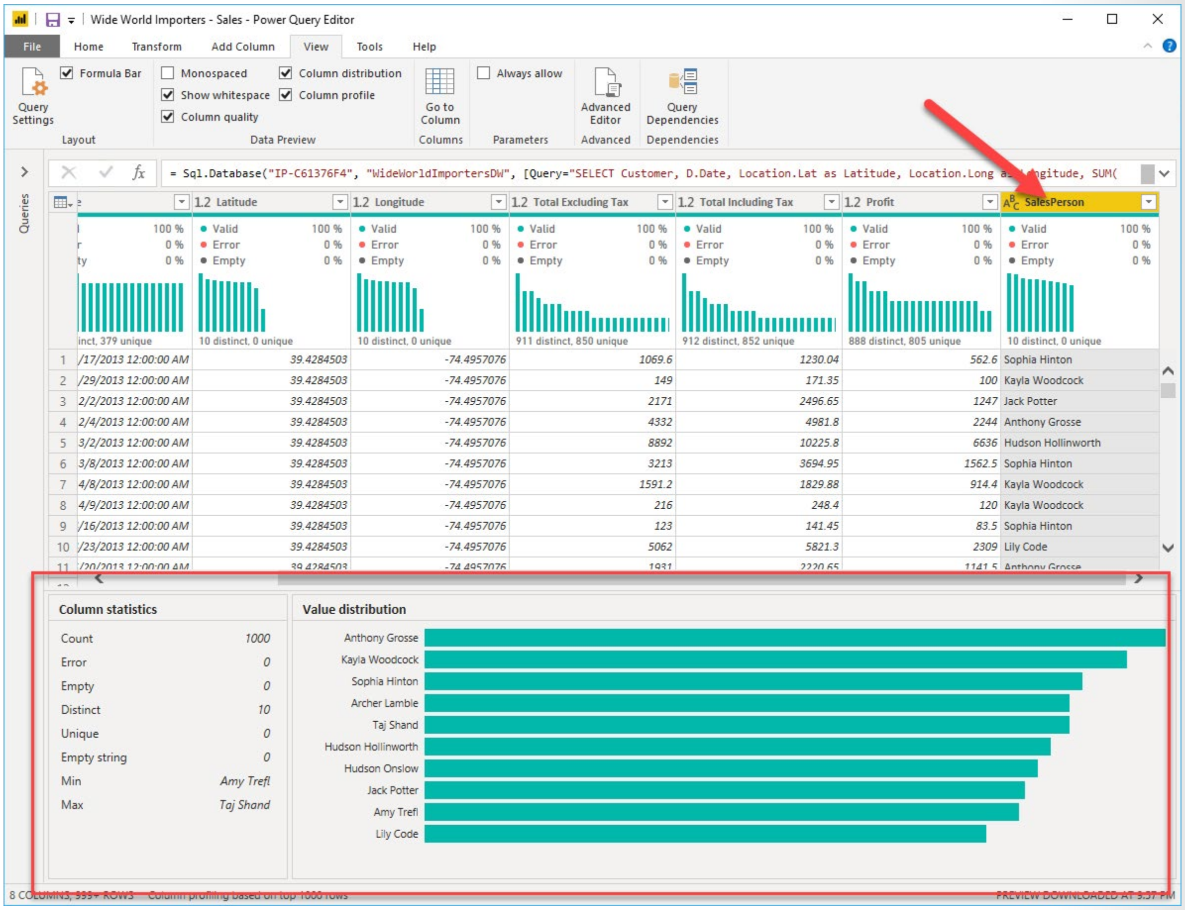
On the **Model** tab, you can edit specific column and table properties by selecting a table or columns, and you can transform the data by using the **Transform Data** button, which takes you to Power Query Editor. Additionally, you can manage, create, edit, and delete relationships between different tables by using **Manage Relationships**, which is located on the ribbon.

Find data anomalies and data statistics

After you have created a connection to a data source and have selected **Transform Data**, you are brought to Power Query Editor, where you can determine if anomalies exist within your data. Data anomalies are outliers within your data. Determining what those anomalies are can help you identify what the normal distribution of your data looks like and whether specific data points exist that you need to investigate further. Power Query Editor determines data anomalies by using the **Column Distribution** feature.

Select **View** on the ribbon, and under **Data Preview**, you can choose from a few options. To understand data anomalies and statistics, select the **Column Distribution**, **Column Quality**, and **Column Profile** options. The following figure shows the statistics that appear.

Column quality and **Column distribution** are shown in the graphs above the columns of data. **Column quality** shows you the percentages of data that is valid, in error, and empty. In an ideal situation, you want 100 percent of the data to be valid.



Column distribution shows you the distribution of the data within the column and the counts of distinct and unique values, both of which can tell you details about the data counts. Distinct values are all values in a column, including duplicates and null values, while unique values do not include duplicates or nulls. Therefore, **distinct** in this table tells you the total count of how many values are present, while **unique** tells you how many of those values are not duplicates or nulls.

Column profile gives you a more in-depth look into the statistics within the column. This column provides several different values, including the count of rows, which is important when verifying whether the importing of your data was successful. For example, if your original database had 100 rows, you could use this row count to verify that 100 rows were, in fact, imported correctly. Additionally, this row count will show how many rows that Power BI has deemed as being outliers (and therefore "errors"), empty rows and strings, and the min and max, which will tell you the smallest and largest value in a column, respectively. This distinction is particularly important in the case of numeric data because it will immediately notify you if you have a maximum value that is beyond what your business identifies as a "maximum." This value calls to your attention these values, which means that you can then focus your efforts when delving deeper into the data. In the case where data was in the text column, as seen in the previous image, the minimum value is the first value and the maximum value is the last value when in alphabetical order.

Additionally, the **Value distribution** graph tells you the counts for each unique value in that specific column. When looking at the graph in the previous image, notice that the value distribution indicates that "Anthony Grosse" appears the greatest number of times within the **SalesPerson** column and that "Lily Code" appears the least amount of times. This information is particularly important because it identifies outliers. If a value appears far more than other values in a column, the **Value distribution** feature allows you to pinpoint a place to begin your investigation into why this is so.

On a numeric column, **Column Statistics** will also include how many zeroes and null values exist, along with the average value in the column, the standard deviation of the values in the column, and how many even and odd values are in the column. These statistics give you an idea of the distribution of data within the column, and are important because they summarize the data in the column and serve as a starting point to determine what the outliers are.

For example, while looking through invoice data, you notice that the **Value distribution** graph shows that a few salespeople in the **SalesPerson** column appear the same amount of times within the data. Additionally, you notice the same situation has occurred in the **Profit** column and in a few other tables as well. During your investigation, you discover that the data you were using was bad data and needed to be refreshed, so you immediately complete the refresh. Without viewing this graph, you might not have seen this error so quickly and, for this reason, value distribution is essential.

After you have completed your edits in Power Query Editor and are ready to begin building visuals, return to **Home** on the Power Query Editor ribbon. Select **Close & Apply**, which will return you to Power BI Desktop and any column edits/transformations will also be applied.

You have now determined the elements that make up profiling data in Power BI, which include loading data in Power BI, interrogating column properties to gain clarity about and make further edits to the type and format of data in columns, finding data anomalies, and viewing data statistics in Power Query Editor. With this knowledge, you can include in your toolkit the ability to study your data in an efficient and effective manner.

Module Review

This module explained how you can take data that is difficult to read, build calculations on, and discover and make it simpler for report authors and others to use. Additionally, you learned how to combine queries so that they were fewer in number, which makes data navigation more streamlined. You also replaced renamed columns into a human readable form and reviewed good naming conventions for objects in Power BI.

Knowledge Check

Question 1

How many rows does Power Query scan to detect the type of data in the columns?

☐ 10,000

☐ 1,000

☐ 100

Question 2

Data profiling is defined as what?

☐ Aggregating columns containing numeric data

☐ Studying the nuances of the data

☐ Data modeling

Question 3

What is the risk of having null values in a numeric column?

☐ DAX expressions that MAX data will be incorrect

☐ DAX expressions that SUM data will be incorrect

☐ DAX expressions that AVERAGE data will be incorrect

Enhance the Data Structure

Apply user-friendly value replacements

When you import data from multiple sources into Power BI Desktop, the data retains its predefined table and column names. You might want to change some of these names so that they are in a consistent format, easier to work with, and more meaningful to a user. You can use Power Query Editor in Power BI Desktop to make these name changes and simplify your data structure.

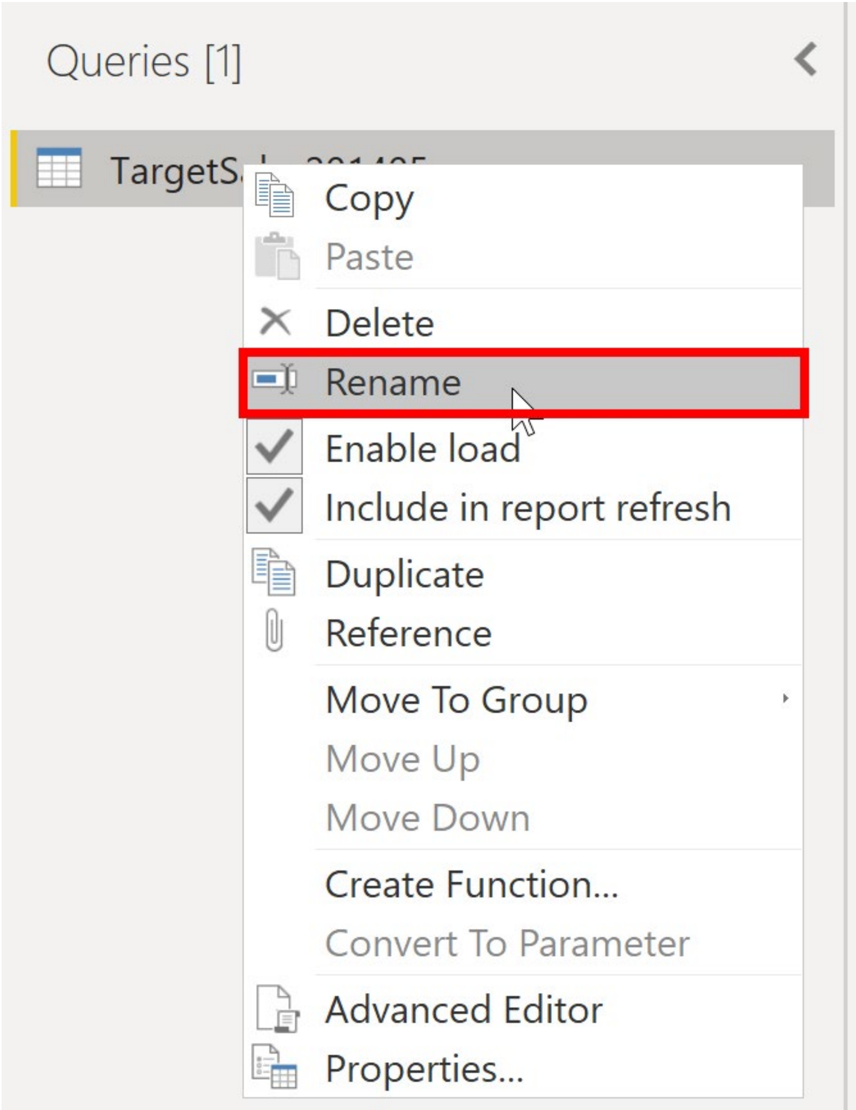
To continue with the previous scenario where you shaped the initial data in your model, you need to take further action to simplify the structure of the sales data and get it ready for developing reports for the Sales team. You have already renamed the columns, but now you need to examine the names of the queries (tables) to determine if any improvements can be made. You also need to review the contents of the columns and replace any values that require correction.

Rename a query

It's good practice to change uncommon or unhelpful query names to names that are more obvious or that the user is more familiar with. For instance, if you import a product fact table into Power BI Desktop and the query name displays as *FactProductTable*, you might want to change it to a more user-friendly name, such as *Products*. Similarly, if you import a view, the view might have a name that contains a prefix of *v*, such as *vProduct*. People might find this name unclear and confusing, so you might want to remove the prefix.

In this example, you have examined the name of the TargetSales query and realize that this name is unhelpful because you'll have a query with this name for every year. To avoid confusion, you want to add the year to the query name.

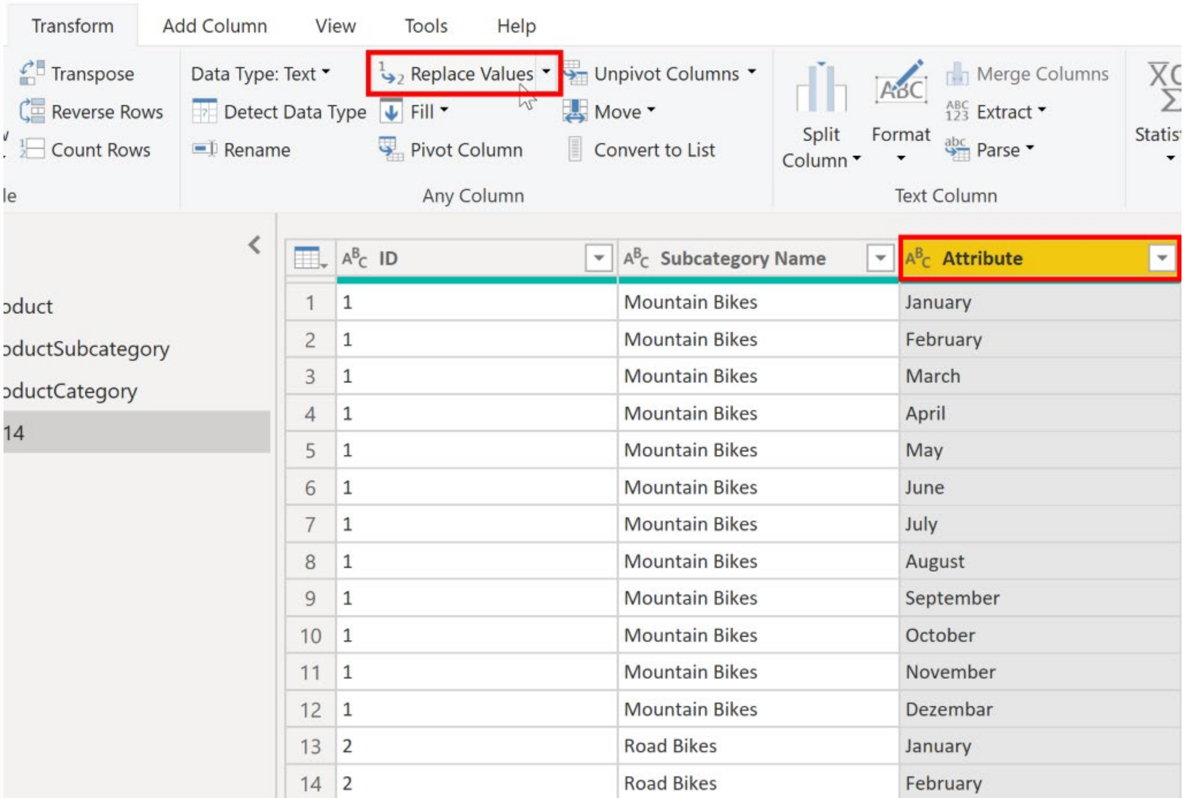
In Power Query Editor, in the **Queries** pane to the left of your data, select the query that you want to rename. Right-click the query and select **Rename**. Edit the current name or type a new name, and then press **Enter**.



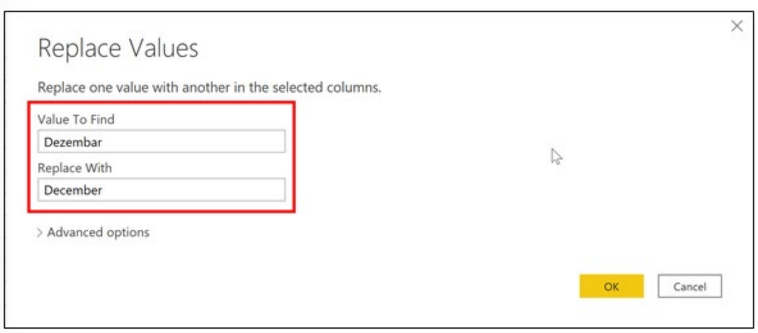
Replace values

You can use the **Replace Values** feature in Power Query Editor to replace any value with another value in a selected column.

In this example, you notice that, in the **Attribute** column, the month December is misspelled. You need to correct this spelling mistake. Select the column that contains the value that you want to replace (**Attribute** in this case), and then select **Replace Values** on the **Transform** tab.



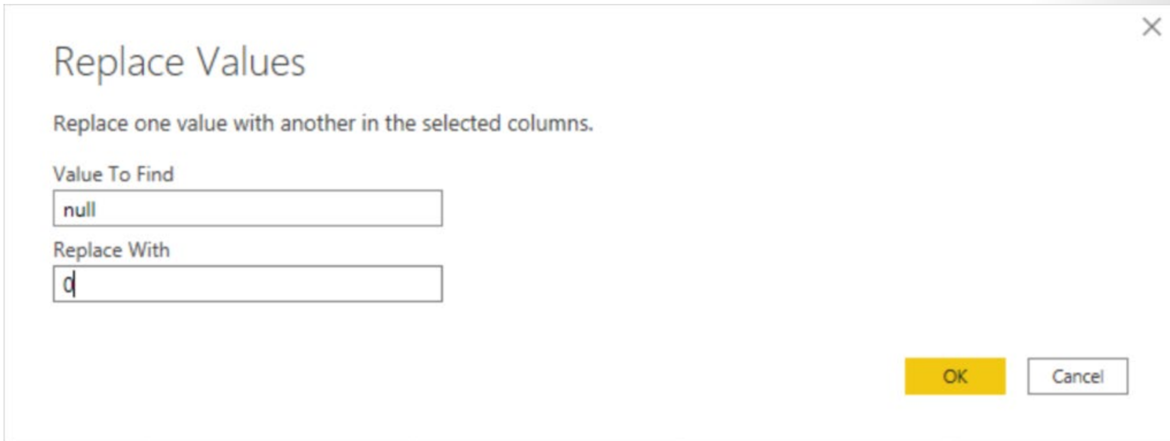
In the **Value to Find** box, enter the name of the value that you want to replace, and then in the **Replace With** box, enter the correct value name and then select **OK**. In Power Query, you can't select one cell and change one value, like you might have done in Excel.



You can review the list of steps that you took to restructure and correct your data in the **Query Settings** pane. When you have completed all steps that you want to take, you can select **Close & Apply** to close Power Query Editor and apply your changes to your data model. However, you can take further action to clean and transform your data.

Replace null values

Occasionally, you might find that your data sources contain null values. For example, a freight amount on a sales order might have a null value if it's synonymous with zero. If the value stays null, theaverages will not calculate correctly. One solution would be to change the nulls to zero, which will produce the more accurate freight average. In this instance, using the same steps that you followed previously will help you replace the null values with zero.

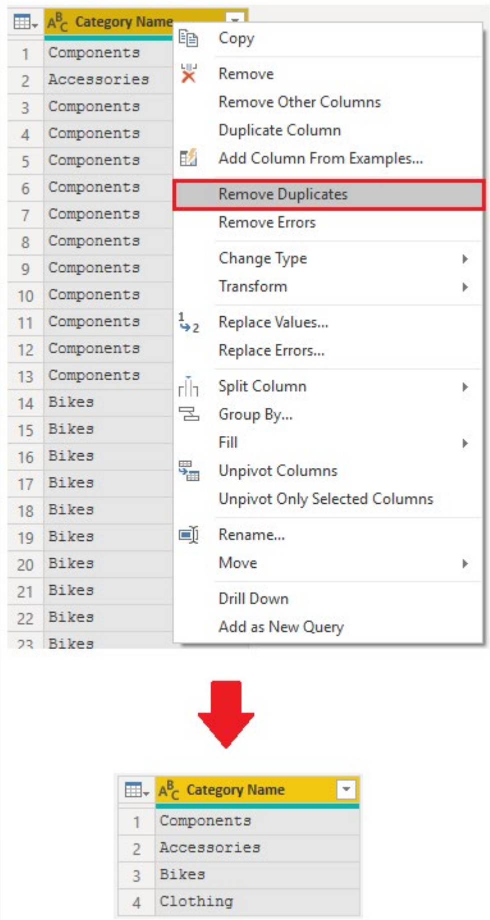


Remove duplicates

You can also remove duplicates from columns to only keep unique names in a selected column by using the **Remove Duplicates** feature in Power Query.

In this example, notice that the **Category Name** column contains duplicates for each category. As a result, you want to create a table with unique categories and use it in your data model. You can achieve this action by selecting a column, right-clicking on the header of the column, and then selecting the **Remove Duplicates** option.

You might consider copying the table before removing the duplicates. The **Copy** option is at the top of the context menu, as shown in the following screenshot. Copying the table before removing duplicates will give you a comparison of the tables and will let you use both tables, if needed.



Best practices for naming tables, columns, and values

Naming conventions for tables, columns, and values have no fixed rules; however, we recommend that you use the language and abbreviations that are commonly used within your organization and that everyone agrees on and considers them as common terminology.

A best practice is to give your tables, columns, and measures descriptive business terms and replace underscores ("_") with spaces. Be consistent with abbreviations, prefaces, and words like “number” and "ID." Excessively short abbreviations can cause confusion if they are not commonly used within the organization.

Also, by removing prefixes or suffixes that you might use in table names and instead naming them in a simple format, you will help avoid confusion.

When replacing values, try to imagine how those values will appear on the report. Values that are too long might be difficult to read and fit on a visual. Values that are too short might be difficult to interpret. Avoiding acronyms in values is also a good idea, provided that the text will fit on the visual.

Evaluate and change column data types

When you import a table from any data source, Power BI Desktop automatically starts scanning the first 1,000 rows (default setting) and tries to detect the type of data in the columns. Some situations might occur where Power BI Desktop does not detect the correct data type. Where incorrect data types occur, you will experience performance issues.

You have a higher chance of getting data type errors when you are dealing with flat files, such as comma-separated values (.CSV) files and Excel workbooks (.XLSX), because data was entered manually into the worksheets and mistakes were made. Conversely, in databases, the data types are predefined when tables or views are created.

A best practice is to evaluate the column data types in Power Query Editor before you load the data into a Power BI data model. If you determine that a data type is incorrect, you can change it. You might also want to apply a format to the values in a column and change the summarization default for a column.

To continue with the scenario where you are cleaning and transforming sales data in preparation for reporting, you now need to evaluate the columns to ensure that they have the correct data type. You need to correct any errors that you identify.

You evaluate the **OrderDate** column. As expected, it contains numeric data, but Power BI Desktop has incorrectly set the column data type to Text. To report on this column, you need to change the data type of this column from Text to Date.

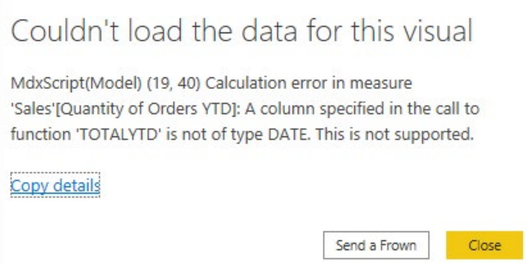
	SalesOrderID	OrderDate	Sort_of_Sales	ProductID	OrderQty
1	52242	07/07/2013	Internet	870	1
2	52592	14/07/2013	Internet	870	1
3	52694	16/07/2013	Internet	870	1
4	52799	18/07/2013	Internet	870	1
5	53799	03/08/2013	Internet	870	1
6	54058	08/08/2013	Internet	870	1
7	54059	08/08/2013	Internet	870	1
8	54063	08/08/2013	Internet	870	1
9	54158	10/08/2013	Internet	870	1
10	54281	12/08/2013	Internet	870	1

Implications of incorrect data types

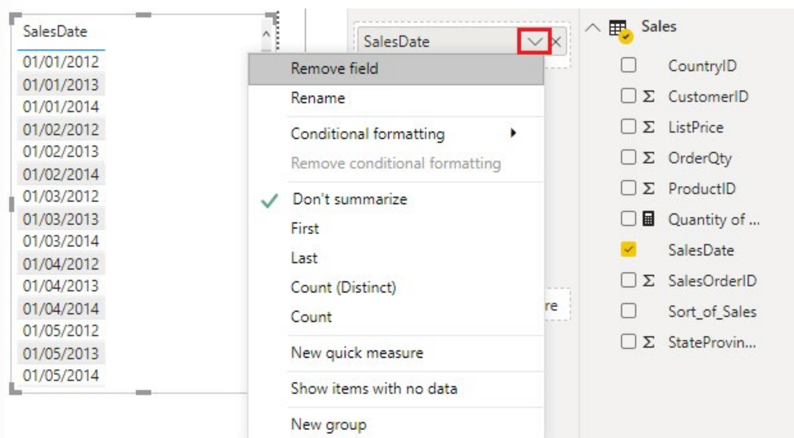
The following information provides insight into problems that can arise when Power BI does not detect the correct data type.

Incorrect data types will prevent you from creating certain calculations, deriving hierarchies, or creating proper relationships with other tables. For example, if you try to calculate the Quantity of Orders YTD, you will get the following error stating that the **OrderDate** column data type is not Date, which is required in time-based calculations.

```
Quantity of Orders YTD = TOTALYTD(SUM('Sales'[OrderQty]), 'Sales'[OrderDate])
```



Another issue with having an incorrect data type applied on a date field is the inability to create a date hierarchy, which would allow you to analyze your data on yearly, monthly, or weekly basis. The following screenshot shows that the SalesDate field is not recognized as type Date and will only be presented as a list of dates in the Table visual. However, it is a best practice to use a date table and turn off the auto date/time to get rid of the auto generated hierarchy. For more information about this process, see **Auto generated data type**[2] documentation.



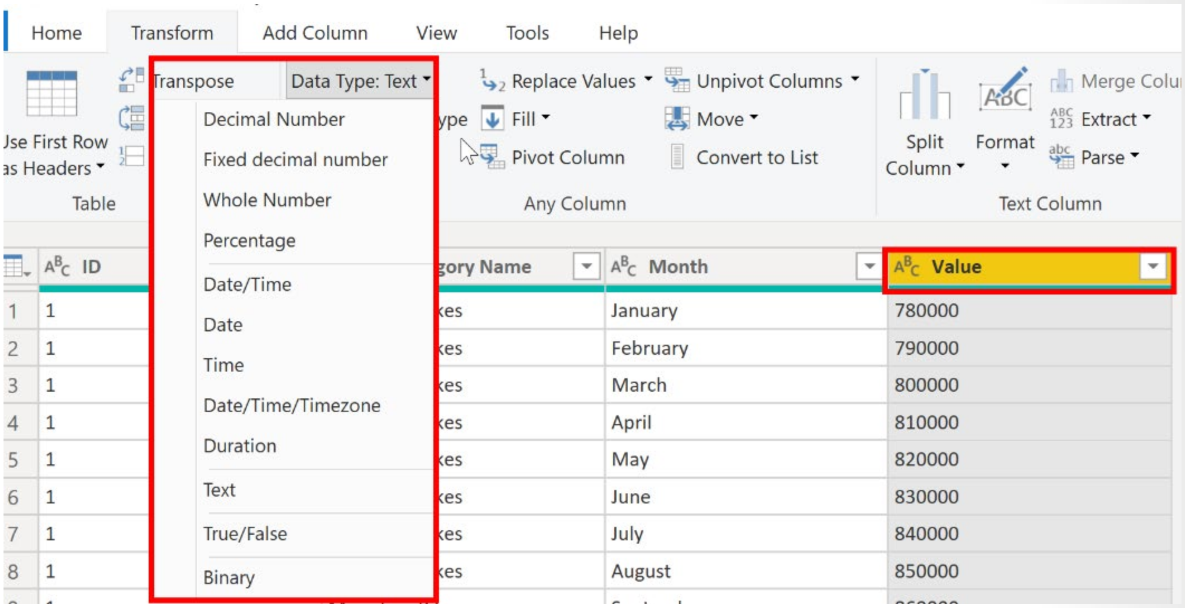
Change the column data type

You can change the data type of a column in two places: in Power Query Editor and in the Power BI Desktop Report view by using the column tools. It is best to change the data type in the Power Query Editor before you load the data.

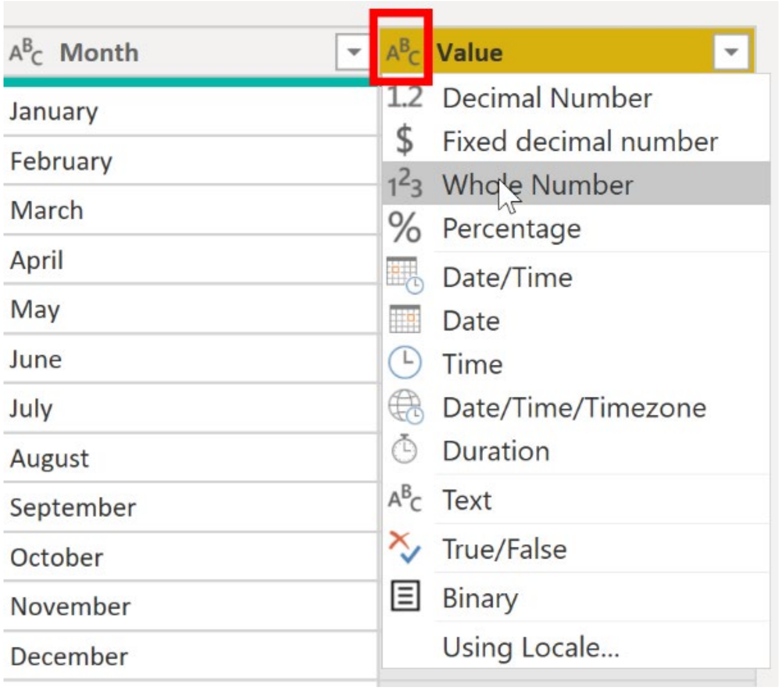
Change the column data type in Power Query Editor

In Power Query Editor, you can change the column data type in two ways. One way is to select the column that has the issue, select **Data Type** in the **Transform** tab, and then select the correct data type from the list.

2 https://docs.microsoft.com/power-bi/guidance/auto-date-time/?azure-portal=true



Another method is to select the data type icon next to the column header and then select the correct data type from the list.



As with any other changes that you make in Power Query Editor, the change that you make to the column data type is saved as a programmed step. This step is called **Changed Type** and it will be iterated every time the data is refreshed.

After you have completed all steps to clean and transform your data, select **Close & Apply** to close Power Query Editor and apply your changes to your data model. At this stage, your data should be in great shape for analysis and reporting.

For more information, see **Data types in Power BI Desktop**[3].

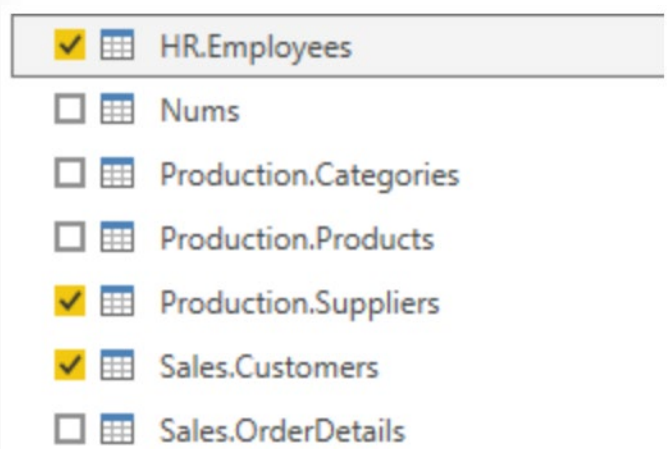
Combine multiple tables into a single table

The ability to combine queries is powerful because it allows you to append or merge different tables or queries together. You can combine tables into a single table in the following circumstances:

- Too many tables exist, making it difficult to navigate an overly-complicated data model.
- Several tables have a similar role.
- A table has only a column or two that can fit into a different table.
- You want to use several columns from different tables in a custom column.

You can combine the tables in two different ways: merging and appending.

Assume that you are developing Power BI reports for the Sales and HR teams. They have asked you to create a contact information report that contains the contact information and location of every employee, supplier, and customer. The data is in the HR.Employees, Production.Suppliers, and the Sales.Customers tables, as shown in the following image.



However, this data comes from multiple tables, so the dilemma is determining how you can merge the data in these multiple tables and create one source-of-truth table to create a report from. The inherent functionality of Power BI allows you to combine and merge queries into a single table.

Append queries

When you append queries, you will be adding rows of data to another table or query. For example, you could have two tables, one with 300 rows and another with 100 rows, and when you append queries, you will end up with 400 rows. When you merge queries, you will be adding columns from one table (or query) into another. To merge two tables, you must have a column that is the key between the two tables.

3 https://docs.microsoft.com/power-bi/connect-data/desktop-data-types/

For the previously mentioned scenario, you will append the HR.Employees table with the Production.Suppliers and Sales.Customers tables so that you have one master list of contact information. Because you want to create one table that has all contact information for employees, suppliers, and customers, when you combine the queries, the pertinent columns that you require in your combined table must be named the same in your original data tables to see one consolidated view.

Before you begin combining queries, you can remove extraneous columns that you don't need for this task from your tables. To complete this task, format each table to have only four columns with your pertinent information, and rename them so they all have the same column headers: ID, company, name, and phone. The following images are snippets of the reformatted Sales.Customers, Production.Suppliers, and HR.Employees tables.

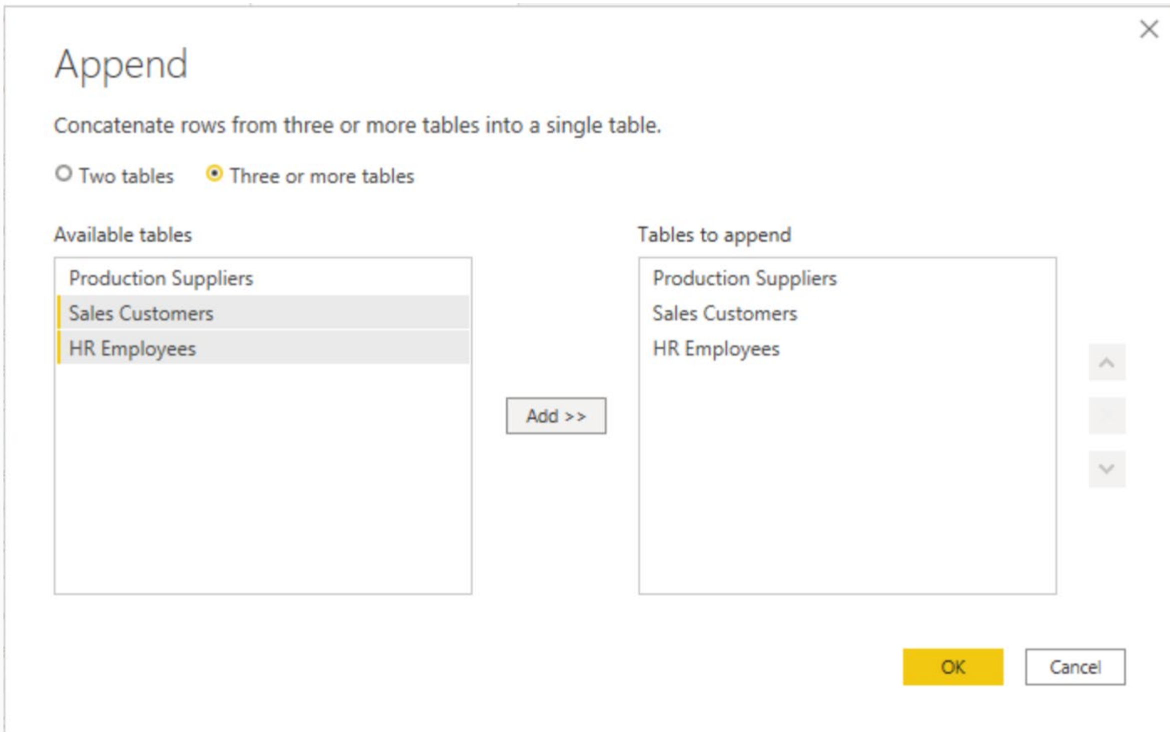
	id	company	name	phone
1	96351	cus145	Mike Poi	1626258811
2	64826	cus134	John Likec	1363718361
3	92647	cus018	Theresa Yulia	2467355553

	id	company	name	phone
1	86563	sup126	Mike Firt	7163615121
2	96352	sup889	Josie Lind	1831635671
3	48256	sup761	Joanna Threven	2936827338

	id	company	name	phone
1	12347	emp124	John Kate	1625535511
2	76478	emp273	Luke John	1028777351
3	82482	emp291	Michael Uji	1245245672

After you have finished reformatting, you can combine the queries. On the **Home** tab on the Power Query Editor ribbon, select the drop-down list for **Append Queries**. You can select **Append Queries as New**, which means that the output of appending will result in a new query or table, or you can select **Append Queries**, which will add the rows from an existing table into another.

Your next task is to create a new master table, so you need to select **Append Queries as New**. This selection will bring you to a window where you can add the tables that you want to append from **Available Tables** to **Tables to Append**, as shown in the following image.



After you have added the tables that you want to append, select **OK**. You will be routed to a new query that contains all rows from all three of your tables, as shown in the following image.

	id	company	name	phone
1	12347	emp124	John Kate	1625535511
2	76478	emp273	Luke John	1028777351
3	82482	emp291	Michael Uji	1245245672
4	97436	emp173	Kate Fitch	2352467634
5	12462	emp270	Eve Jun	3578999554
6	35237	emp715	Don Potre	3579006677
7	23467	emp183	Marc Wetb	2245789954
8	13892	emp163	Sara Scotts	2388367234
9	56356	emp172	Mitch Potter	1234683673
10	23478	emp812	Liliy Kithc	4567800522
11	45783	emp818	Ren Swrte	2357997515
12	86563	sup126	Mike Firt	7163615121
13	96352	sup889	Josie Lind	1831635671
14	48256	sup761	Joanna Threven	2936827338
15	28461	sup163	Michael Bob	1937293165
16	83613	sup162	Mimi Jukth	2916384462
17	96351	cus145	Mike Poi	1626258811
18	64826	cus134	John Likec	1363718361
19	92647	cus018	Theresa Yulia	2467355553
20	91661	cus182	Ren Thibe	3345783234
21	1736	cus104	Ron Mikel	1235799789
22	1835	cus103	Joy Qui	2345689411
23	1745	cus141	Cat Yate	2345678986

You have now succeeded in creating a master table that contains the information for the employees, suppliers, and customers. You can exit Power Query Editor and build any report elements surrounding this master table.

However, if you wanted to merge tables instead of appending the data from one table to another, the process would be different.

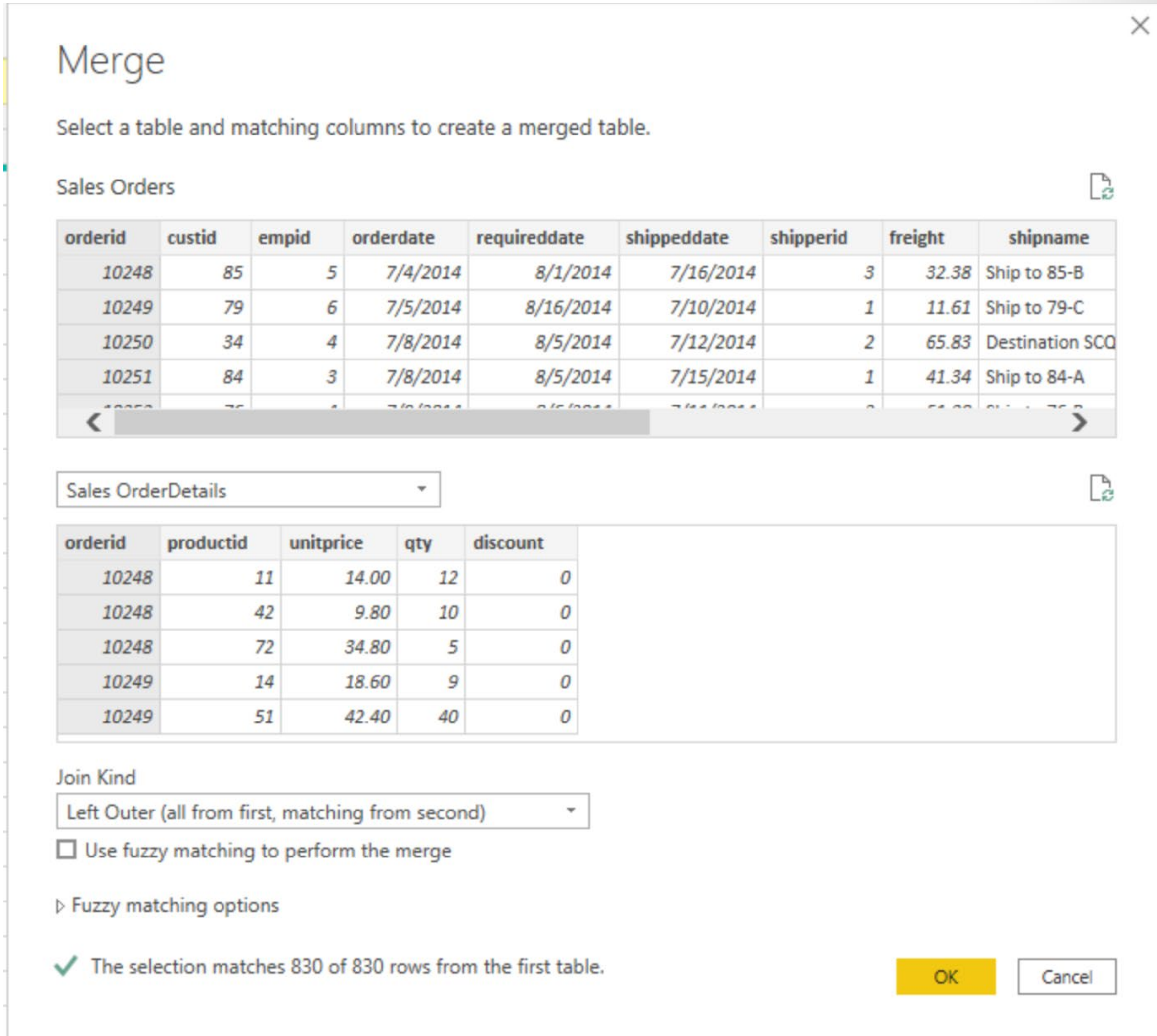
Merge queries

When you merge queries, you are combining the data from multiple tables into one based on a column that is common between the tables. This process is similar to the JOIN clause in SQL. Consider a scenario where the Sales team now wants you to consolidate orders and their corresponding details (which are currently in two tables) into a single table. You can accomplish this task by merging the two tables, Orders and OrderDetails, as shown in the following image. The column that is shared between these two tables is **OrderID**.

	orderid	orderdate	shipperid
1	1	4/23/2018	12
2	2	4/25/2018	24
3	3	6/12/2018	19
4	4	6/13/2018	13
5	5	7/23/2018	11
6	6	7/25/2018	33

	orderid	productid	qty	unitprice
1	1	124	12	14
2	2	134	55	11.2
3	3	641	57	45
4	4	98	5	112.5
5	5	312	23	11.1
6	6	124	78	11.2

Go to **Home** on the Power Query Editor ribbon and select the **Merge Queries** drop-down menu, where you can select **Merge Queries as New**. This selection will open a new window, where you can choose the tables that you want to merge from the drop-down list, and then select the column that is matching between the tables, which in this case is **orderid**.



You can also choose how to join the two tables together, a process that is also similar to JOIN statements in SQL. These join options include:

- **Left Outer** - Displays all rows from the first table and only the matching rows from the second.
- **Full Outer** - Displays all rows from both tables.
- **Inner** - Displays the matched rows between the two tables.

For this scenario, you will choose to use a **Left Outer** join. Select **OK**, which will route you to a new window where you can view your merged query.

	orderid	orderdate	shipperid	OrderDetails.productid	OrderDetails.qty	OrderDetails.unitprice
1	1	4/23/2018	12	124	12	14
2	2	4/25/2018	24	134	55	11.2
3	3	6/12/2018	19	641	57	45
4	4	6/13/2018	13	98	5	112.5
5	5	7/23/2018	11	312	23	11.1
6	6	7/25/2018	33	124	78	11.2
7	7	8/1/2019	77	137	11	572.1
8	8	8/10/2019	11	124	36	1331.9
9	9	8/11/2019	81	789	85	898.1

Now, you can merge two queries or tables in different ways so that you can view your data in the most appropriate way for your business requirements.

For more information on this topic, see the **Shape and Combine Data in Power BI**[4] documentation.

Knowledge Check

Question 1

What is not a best practice for naming conventions in Power BI?

- ☐ Rename columns to have spaces in them
- ☐ Replace values that have integers with human readable results
- ☐ Abbreviated column names

Question 2

If you have two queries that contain different data with the same structure, and you want to combine them into one query, which operation should you perform?

- ☐ Merge
- ☐ Append
- ☐ Combine column

4 https://docs.microsoft.com/power-bi/connect-data/desktop-shape-and-combine-data/

Answers

Question 1

The primary data preparation tool in Power BI is called what?

- ☐ Report editor
- ■ Power Query editor
- ☐ Data editor

Question 2

The process of shaping data by converting your flat data into a table that contains an aggregation value for each unique value in a column is called what?

- ■ Group by columns
- ☐ Pivot (pivoting a column)
- ☐ Manage aggregations

Question 3

What can be achieved by removing unnecessary rows and columns?

- ☐ It is not necessary to delete unnecessary rows and columns and it is a good practice to keep all metadata intact.
- ☐ Deleting unnecessary rows and columns can damage the structure of the data model.
- ■ Deleting unnecessary rows and columns will reduce the dataset size and it is a good practice to load only necessary data into your data model.

Question 1

How many rows does Power Query scan to detect the type of data in the columns?

- ☐ 10,000
- ■ 1,000
- ☐ 100

Question 2

Data profiling is defined as what?

- ☐ Aggregating columns containing numeric data
- ■ Studying the nuances of the data
- ☐ Data modeling

Question 3

What is the risk of having null values in a numeric column?

- ☐ DAX expressions that MAX data will be incorrect
- ☐ DAX expressions that SUM data will be incorrect
- ■ DAX expressions that AVERAGE data will be incorrect

Question 1

What is not a best practice for naming conventions in Power BI?

- ☐ Rename columns to have spaces in them
- ☐ Replace values that have integers with human readable results
- ■ Abbreviated column names

Question 2

If you have two queries that contain different data with the same structure, and you want to combine them into one query, which operation should you perform?

- ☐ Merge
- ■ Append
- ☐ Combine column

Module 4 Design a Data Model in Power BI

Introduction to Data Modeling

Introduction to data modeling

Creating a great data model is one of the most important tasks that a data analyst can perform in Microsoft Power BI. By doing this job well, you help make it easier for people to understand your data, which will make building valuable Power BI reports easier for them and for you.

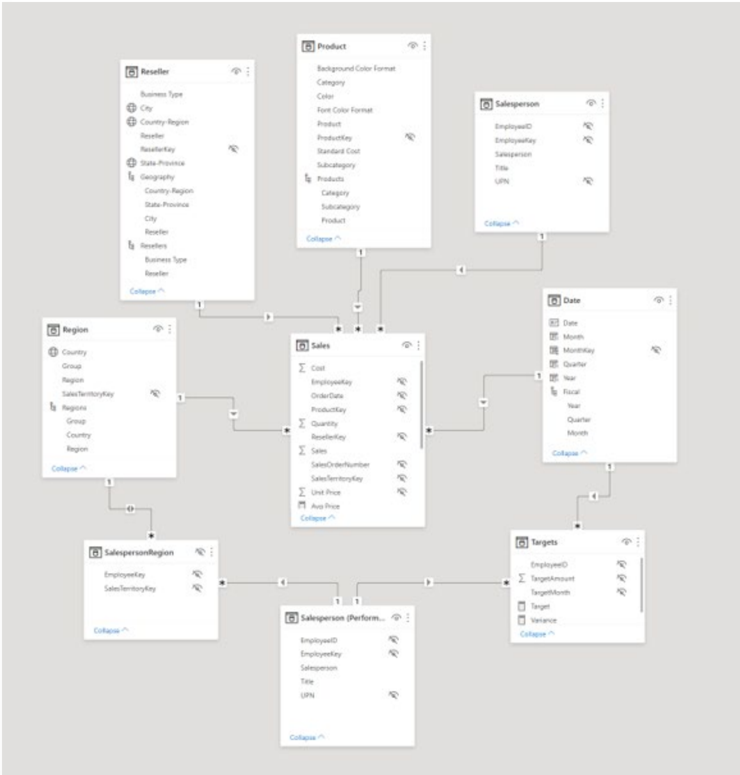
A good data model offers the following benefits:

- Data exploration is faster.
- Aggregations are simpler to build.
- Reports are more accurate.
- Writing reports takes less time.
- Reports are easier to maintain in the future.

Providing set rules for what makes a good data model is difficult because all data is different, and the usage of that data varies. Generally, a smaller data model is better because it will perform faster and will be simpler to use. However, defining what a smaller data model entails is equally as problematic because it's a heuristic and subjective concept.

Typically, a smaller data model is comprised of fewer tables and fewer columns in each table that the user can see. If you import all necessary tables from a sales database, but the total table count is 30 tables, the user will not find that intuitive. Collapsing those tables into five tables will make the data model more intuitive to the user, whereas if the user opens a table and finds 100 columns, they might find it overwhelming. Removing unneeded columns to provide a more manageable number will increase the likelihood that the user will read all column names. To summarize, you should aim for simplicity when designing your data models.

The following image is an example data model. The boxes contain tables of data, where each line item within the box is a column. The lines that connect the boxes represent relationships between the tables. These relationships can be complex, even in such a simplistic model. The data model can become easily disorganized, and the total table count in the model can gradually increase. Keeping your data model simple, comprehensive, and accurate requires constant effort.



Power BI allows relationships to be built from tables with different data sources, a powerful function that enables you to pull one table from Microsoft Excel and another from a relational database. You would then create the relationship between those two tables and treat them as a unified dataset.

Now that you have learned about the relationships that make up the data schema, you'll be able to explore a specific type of schema design, the star schema, which is optimized for high performance and usability.

Joins and relationships

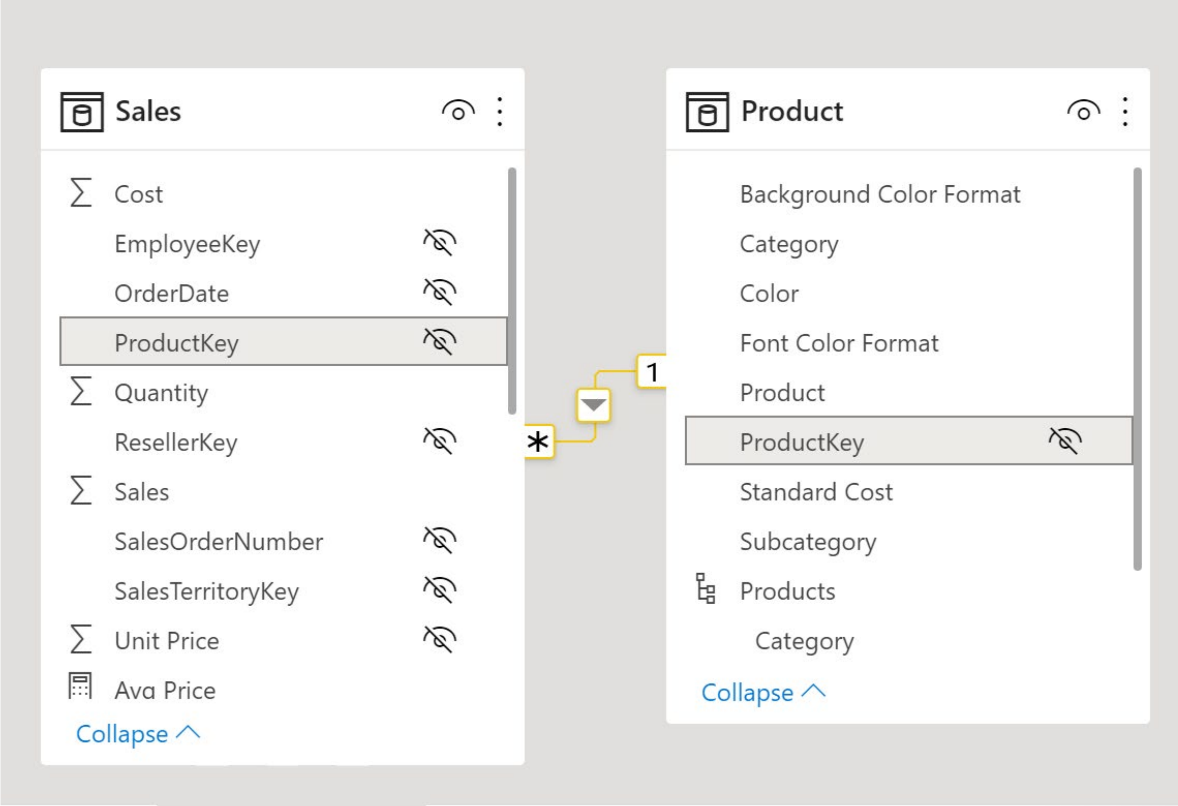
Primary Keys and Foreign Keys

Relationships are defined between tables through primary and foreign keys. Primary keys are column(s) that identify each unique, non-null data row. For instance, if you have a Customers table, you could have an index that identifies each unique customer. The first row will have an ID of 1, the second row an ID of 2, and so on. Each row is assigned a unique value, which can be referred to by this simple value: the primary key. This process becomes important when you are referencing rows in a different table, which is what foreign keys do. Relationships between tables are formed when you have primary and foreign keys in common between different tables.

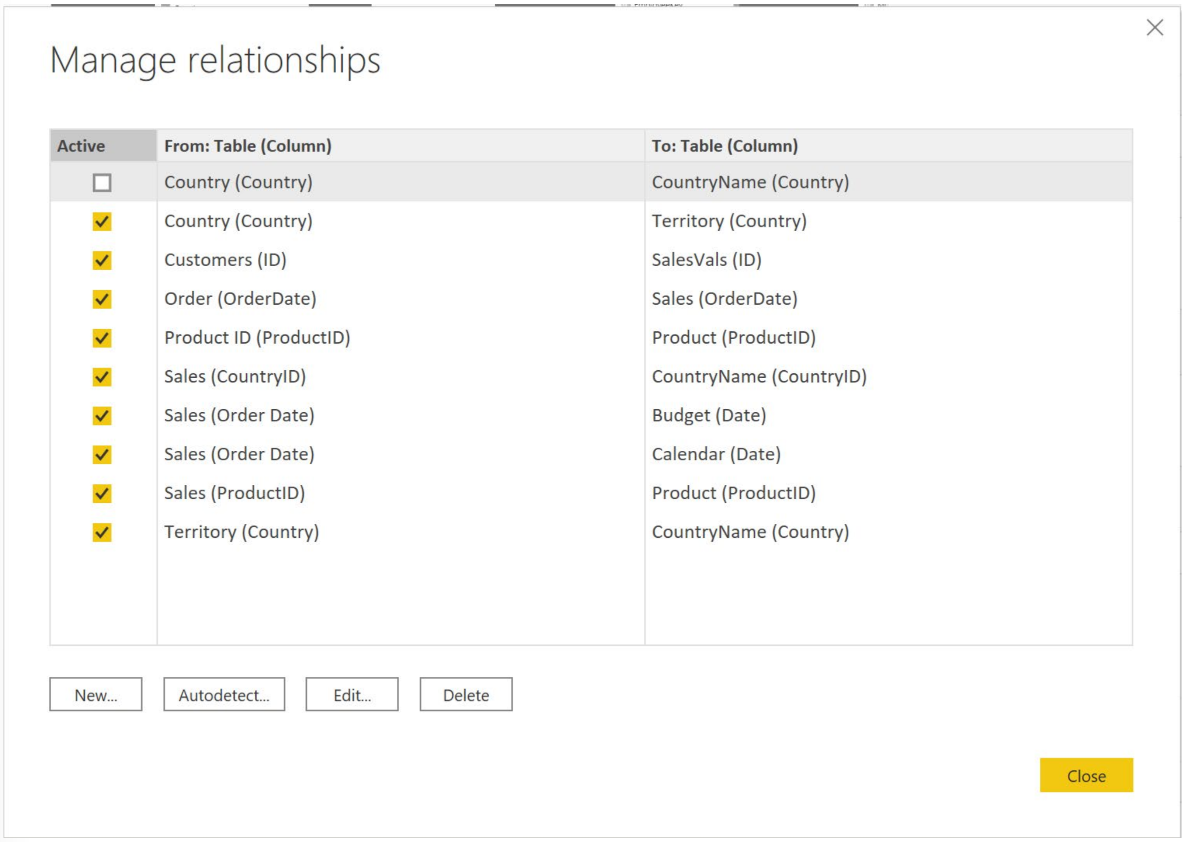
Power BI allows relationships to be built from tables with different data sources, a powerful function that enables you to pull one table from Microsoft Excel and another from a relational database. You would then create the relationship between those two tables and treat them as a unified dataset.

Relationships

Assuming that you've already retrieved your data and cleaned it in Power Query, you can then go to the **Model** tab, where the data model is located. The following image shows how the relationship between the **Sales** and **Product** tables can be seen through the **ProductKey** column.



To manage these relationships, go to **Manage Relationships** on the ribbon, where the following window will appear.



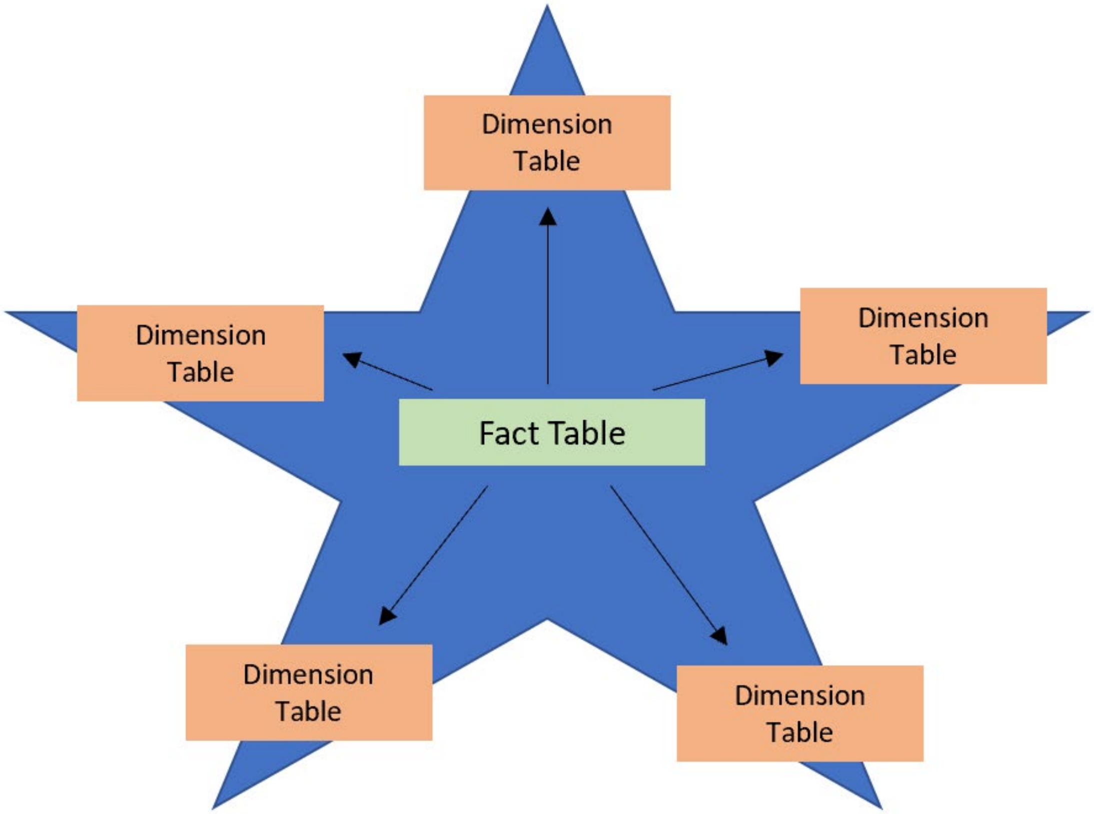
Active	From: Table (Column)	To: Table (Column)
☐	Country (Country)	CountryName (Country)
☑	Country (Country)	Territory (Country)
☑	Customers (ID)	SalesVals (ID)
☑	Order (OrderDate)	Sales (OrderDate)
☑	Product ID (ProductID)	Product (ProductID)
☑	Sales (CountryID)	CountryName (CountryID)
☑	Sales (Order Date)	Budget (Date)
☑	Sales (Order Date)	Calendar (Date)
☑	Sales (ProductID)	Product (ProductID)
☑	Territory (Country)	CountryName (Country)

In this view, you can create, edit, and delete relationships between tables and also autodetect relationships that already exist. When you load your data into Power BI, the **Autodetect** feature will help you establish relationships between columns that are named similarly. Relationships can be inactive or active. Only one active relationship can exist between tables, which is discussed in a future module.

While the **Manage Relationships** feature allows you to configure relationships between tables, you can also configure table and column properties to ensure organization in your table structure.

Star schemas

You can design a star schema to simplify your data. It's not the only way to simplify your data, but it is a popular method; therefore, every Power BI data analyst should understand it. In a star schema, each table within your dataset is defined as a dimension or a fact table, as shown in the following visual.



Fact tables contain observational or event data values: sales orders, product counts, prices, transactional dates and times, and quantities. Fact tables can contain several repeated values. For example, one product can appear multiple times in multiple rows, for different customers on different dates. These values can be aggregated to create visuals. For instance, a visual of the total sales orders is an aggregation of all sales orders in the fact table. With fact tables, it is common to see columns that are filled with numbers and dates. The numbers can be units of measurement, such as sale amount, or they can be keys, such as a customer ID. The dates represent time that is being recorded, like order date or shipped date.

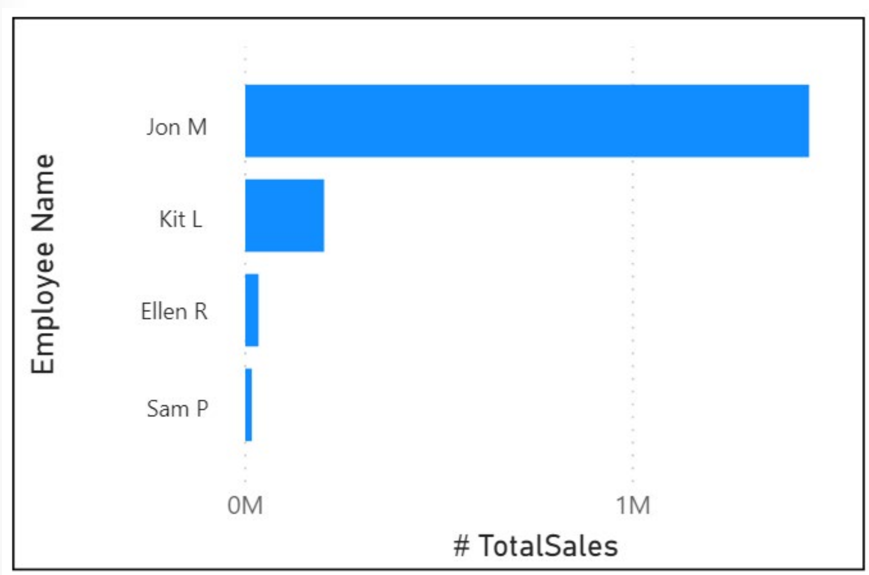
Dimension tables contain the details about the data in fact tables: products, locations, employees, and order types. These tables are connected to the fact table through key columns. Dimension tables are used to filter and group the data in fact tables. The dimension tables, by contrast, contain unique values, for instance, one row for each product in the Products table and one row for each customer in the Customer table. For the total sales orders visual, you could group the data so that you see total sales orders by product, in which product is data in the dimension table.

Fact tables are usually much larger than dimension tables because numerous events occur in fact tables, such as individual sales. Dimension tables are typically smaller because you are limited to the number of items that you can filter and group on. For instance, a year contains only so many months, and the United States is comprised of only a certain number of states.

Considering this information about fact tables and dimension tables, you might wonder how you can build this visual in Power BI.

The pertinent data resides in two tables, Employee and Sales, as shown in the following data model. Because the Sales table contains the sales order values, which can be aggregated, it is considered a fact table. The Employee table contains the specific employee name, which filters the sales orders, so it would be a dimension table. The common column between the two tables, which is the primary key in the Employee table, is **EmployeeID**, so you can establish a relationship between the two tables based on this column.

When creating this relationship, you can build the visual according to the requirements, as shown in the following figure. If you did not establish this relationship, while keeping in mind the commonality between the two tables, you would have had more difficulty building your visual.



Star schemas and the underlying data model are the foundation of organized reports; the more time you spend creating these connections and design, the easier it will be to create and maintain reports.

Knowledge Check

Question 1

The two types of tables in a star schema are what?

☐ Active and inactive tables

☐ Qualitative and quantitative data tables

☐ Fact and dimension tables

Question 2

What is the difference between a fact table and a dimension table?

☐ Fact tables store observations or events while dimension tables contain information about specific entities within the data

☐ Fact tables contain information about specific entities while dimension tables contain information about observational data

☐ Dimension tables tell you about specific roles in Power BI while fact tables tell you information about facts that are associated with those roles in Power BI

☐ There is no difference

Working with Tables

Define and configure table and column properties

When users see fewer tables, they will enjoy using your data model considerably more. For example, suppose you've imported dozens of tables from many data sources and now the visual appears disorderly. In this case, you need to ensure that, before you begin working on building reports, your data model and table structure are simplified.

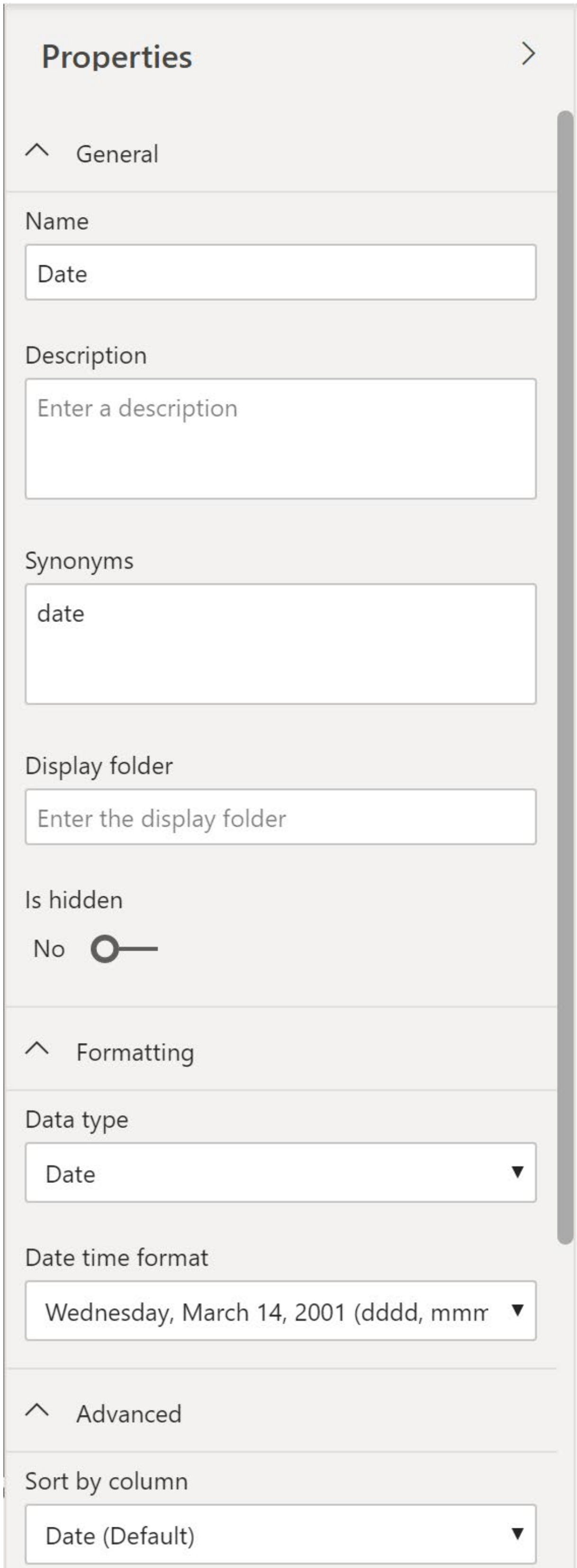
A simple table structure will:

- Be simple to navigate because of column and table properties that are specific and user-friendly.
- Have merged or appended tables to simplify the tables within your data structure.
- Have good-quality relationships between tables that make sense.

The following sections further explain how you might work with your tables to ensure a simple and readable table structure.

Configure table and column properties

The **Model** view in Power BI desktop provides many options within the column properties that you can view or update. A simple method to get to this menu to update the tables and fields is by Ctrl+clicking or Shift+clicking items on this page.


Properties
General
Name
Date
Description
Enter a description
Synonyms
date
Display folder
Enter the display folder
Is hidden
No
Formatting
Data type
Date
Date time format
Wednesday, March 14, 2001 (dddd, mmm
Advanced
Sort by column
Date (Default)

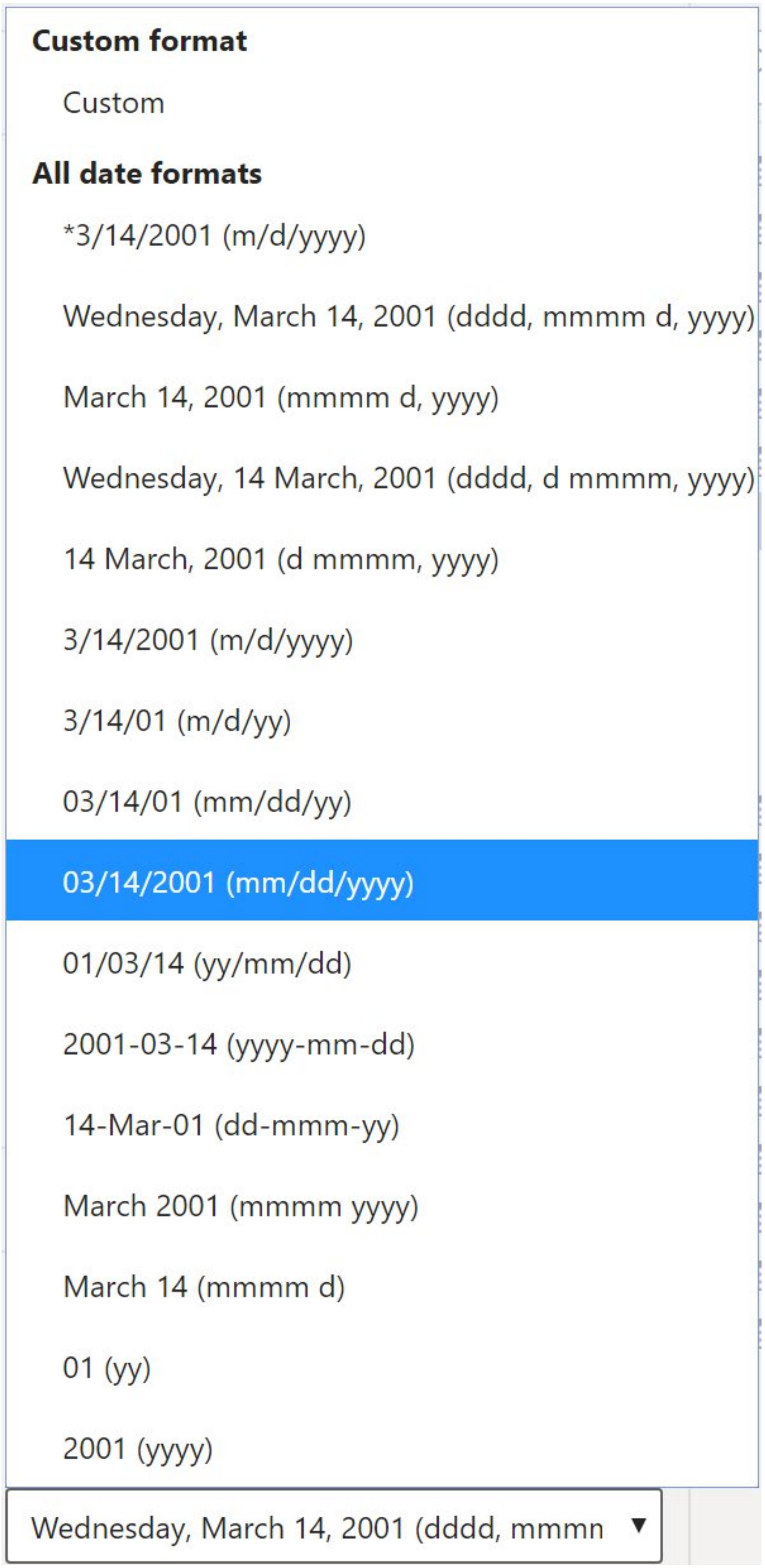
Under the **General** tab, you can:

- Edit the name and description of the column.
- Add synonyms that can be used to identify the column when you are using the Q&A feature.
- Add a column into a folder to further organize the table structure.
- Hide or show the column.

Under the **Formatting** tab, you can:

- Change the data type.
- Format the date.

For instance, suppose that the dates in your column are formatted, as seen in the previous screenshot, in the form of “Wednesday, March 14, 2001”. If you want to change the format so that the date was in the "mm/dd/yyyy" format, you would select the drop-down menu under **All date time formats** and then choose the appropriate date format, as shown in the following figure.



After selecting the appropriate date format, return to the **Date** column, where you should see that the format has indeed changed, as shown in the following figure.

Date
01/01/2019
01/06/2019
01/13/2019
01/20/2019
01/27/2019
02/03/2019
02/10/2019

Under the **Advanced** tab, you can:

- Sort by a specific column.
- Assign a specific category to the data.
- Summarize the data.
- Determine if the column or table contains null values.

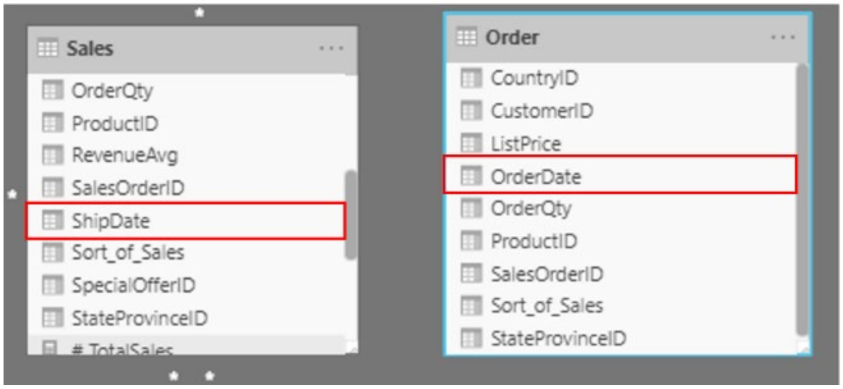
Additionally, Power BI has a new functionality to update these properties on many tables and fields by Ctrl+clicking or Shift+clicking items.

These examples are only some of the many types of transformations that you can make to simplify the table structure. This step is important to take before you begin making your visuals so that you don't have to go back and forth when making formatting changes. This process of formatting and configuring tables can also be done in Power Query.

Create a date table

During report creation in Power BI, a common business requirement is to make calculations based on date and time. Organizations want to know how their business is doing over months, quarters, fiscal years, and so on. For this reason, it is crucial that these time-oriented values are formatted correctly. Power BI autodetects for date columns and tables; however, situations can occur where you will need to take extra steps to get the dates in the format that your organization requires.

For example, suppose that you are developing reports for the Sales team at your organization. The database contains tables for sales, orders, products, and more. You notice that many of these tables, including Sales and Orders, contain their own date columns, as shown by the **ShipDate** and **OrderDate** columns in the Sales and Orders tables. You are tasked with developing a table of the total sales and orders by year and month. How can you build a visual with multiple tables, each referencing their own date columns?



To solve this problem, you can create a common date table that can be used by multiple tables. The following section explains how you can accomplish this task in Power BI.

Create a common date table

Ways that you can build a common date table are:

- Source data
- DAX
- Power Query

Source data

Occasionally, source databases and data warehouses already have their own date tables. If the administrator who designed the database did a thorough job, these tables can be used to perform the following tasks:

- Identify company holidays
- Separate calendar and fiscal year
- Identify weekends versus weekdays

Source data tables are mature and ready for immediate use. If you have a table as such, bring it into your data model and don't use any other methods that are outlined in this section. We recommend that you use a source date table because it is likely shared with other tools that you might be using in addition to Power BI.

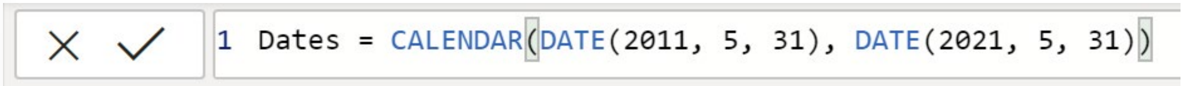
If you do not have a source data table, you can use other ways to build a common date table.

DAX

You can use the Data Analysis Expression (DAX) functions CALENDARAUTO() or CALENDAR() to build your common date table. The CALENDAR() function returns a contiguous range of dates based on a start and end date that are entered as arguments in the function. Alternatively, the CALENDARAUTO() function returns a contiguous, complete range of dates that are automatically determined from your dataset. The starting date is chosen as the earliest date that exists in your dataset, and the ending date is the latest date that exists in your dataset plus data that has been populated to the fiscal month that you can choose to include as an argument in the CALENDARAUTO() function. For the purposes of this example, the CALENDAR() function is used because you only want to see the data from May 31, 2011 (the first day that Sales began its tracking of this data) and forward for the next 10 years.

In Power BI Desktop, go to the **Modeling** tab on the ribbon. Select **New Table**, and then enter in the following DAX formula:

```
Dates  = CALENDAR(DATE(2011, 5, 31), DATE(2021, 5, 31))
```



Now, you have a column of dates that you can use. However, this column is slightly sparse. You also want to see columns for just the year, the month number, the week of the year, and the day of the week. You can accomplish this task by selecting **New Column** on the ribbon and entering the following DAX equation, which will retrieve the year from your Date table.

```
Year = YEAR(Dates[Date])
```

Date	Year
Tuesday, May 31, 2011	2011
Wednesday, June 1, 2011	2011
Thursday, June 2, 2011	2011
Friday, June 3, 2011	2011
Saturday, June 4, 2011	2011

You can perform the same process to retrieve the month number, week number, and day of the week:

```
MonthNum = MONTH(Dates[Date])

WeekNum = WEEKNUM(Dates[Date])

DayoftheWeek = FORMAT(Dates[Date].[Day], "DDDD")
```

When you have finished, your table will contain the columns that are shown in the following figure.

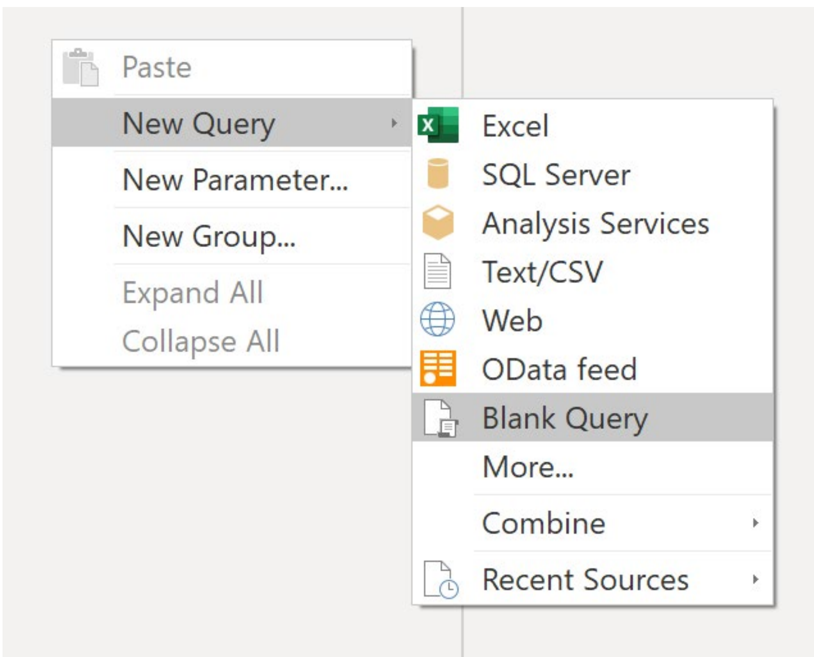
Date	Year	MonthNum	WeekNum	DayoftheWeek
Tuesday, May 31, 2011	2011	5	23	Tuesday
Wednesday, June 1, 2011	2011	6	23	Sunday
Thursday, June 2, 2011	2011	6	23	Monday
Friday, June 3, 2011	2011	6	23	Tuesday

You have now created a common date table by using DAX. This process only adds your new table to the data model; you will still need to establish relationships between your date table and the Sales and Order tables, and then mark your table as the official date table of your data model. However, before you complete those tasks, make sure that you consider another way of building a common date table: by using Power Query.

Power Query

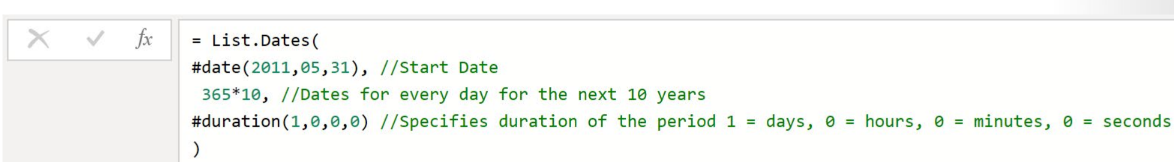
You can also use M-language, the development language that is used to build queries in Power Query, to define a common date table.

Select **Transform Data** in Power BI Desktop, which will direct you to Power Query. In the blank space of the left **Queries** pane, right-click to open the following drop-down menu, where you will select **New Query > Blank Query**.



In the resulting **New Query** view, enter the following M-formula to build a calendar table:

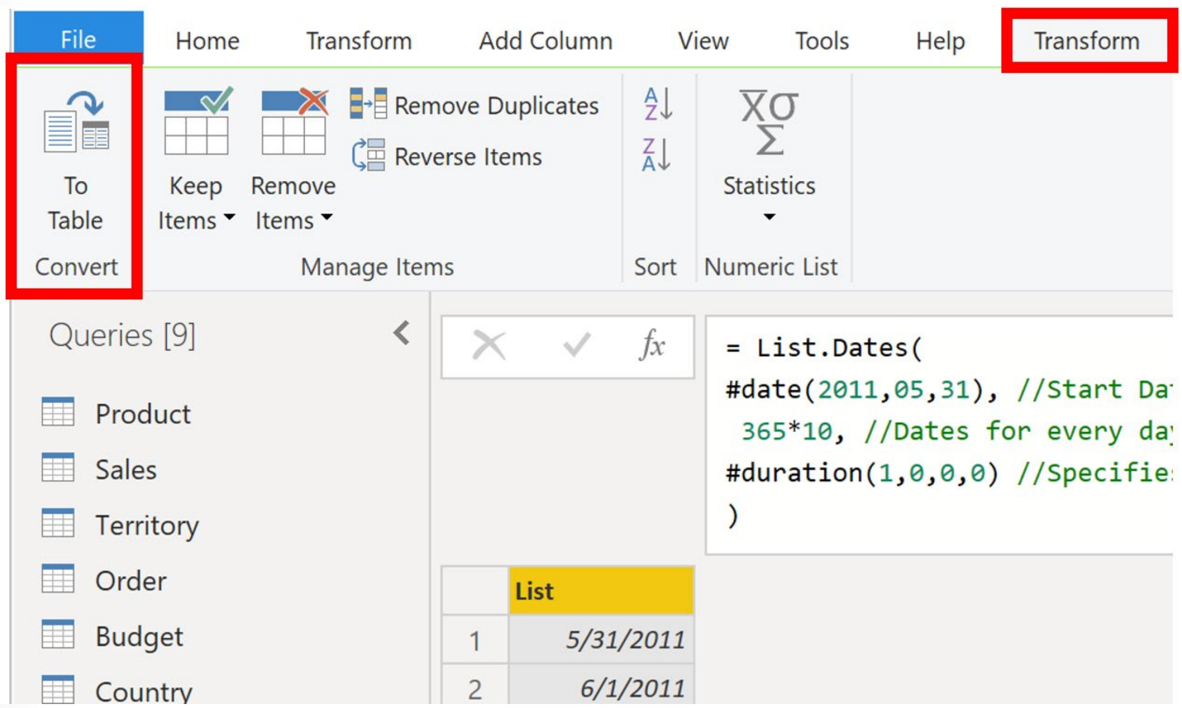
```
= List.Dates(#date(2011,05,31), 365*10, #duration(1,0,0,0)
```



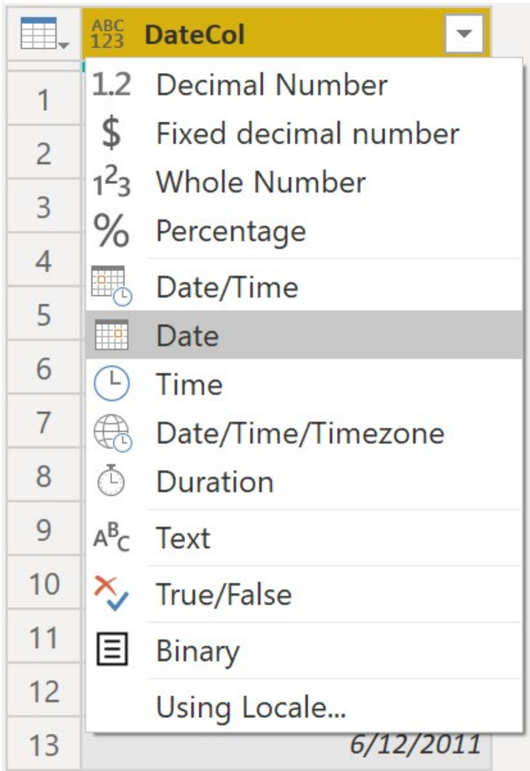
For your sales data, you want the start date to reflect the earliest date that you have in your data: May 31, 2011. Additionally, you want to see dates for the next 10 years, including dates in the future. This approach ensures that, as new sales data flows in, you won't have to re-create this table. You can also change duration. In this case, you want a data point for every day, but you can also increment by hours, minutes, and seconds. The following figure shows the result.

	List
1	5/31/2011
2	6/1/2011
3	6/2/2011
4	6/3/2011
5	6/4/2011
6	6/5/2011
7	6/6/2011
8	6/7/2011
9	6/8/2011

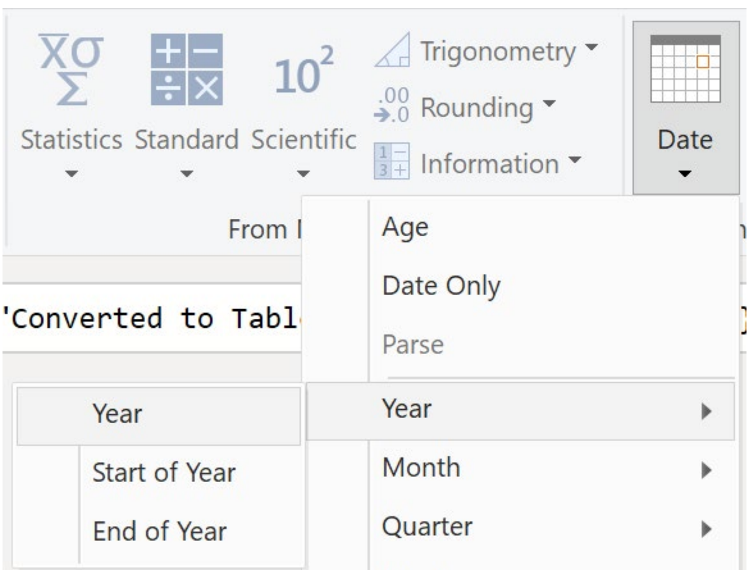
After you have realized success in the process, you notice that you have a list of dates instead of a table of dates. To correct this error, go to the **Transform** tab on the ribbon and select **Convert > To Table**. As the name suggests, this feature will convert your list into a table. You can also rename the column to **DateCol**.



Next, you want to add columns to your new table to see dates in terms of year, month, week, and day so that you can build a hierarchy in your visual. Your first task is to change the column type by selecting the icon next to the name of the column and, in the resulting drop-down menu, selecting the **Date** type.



After you have finished selecting the **Date** type, you can add columns for year, months, weeks, and days. Go to **Add Column**, select the drop-down menu under **Date**, and then select **Year**, as shown in the following figure.



Notice that Power BI has added a column of all years that are pulled from **DateCol**.

DateCol	Year
5/31/2011	2011
6/1/2011	2011
6/2/2011	2011

Complete the same process for months, weeks, and days. After you have finished this process, the table will contain the columns that are shown in the following figure.

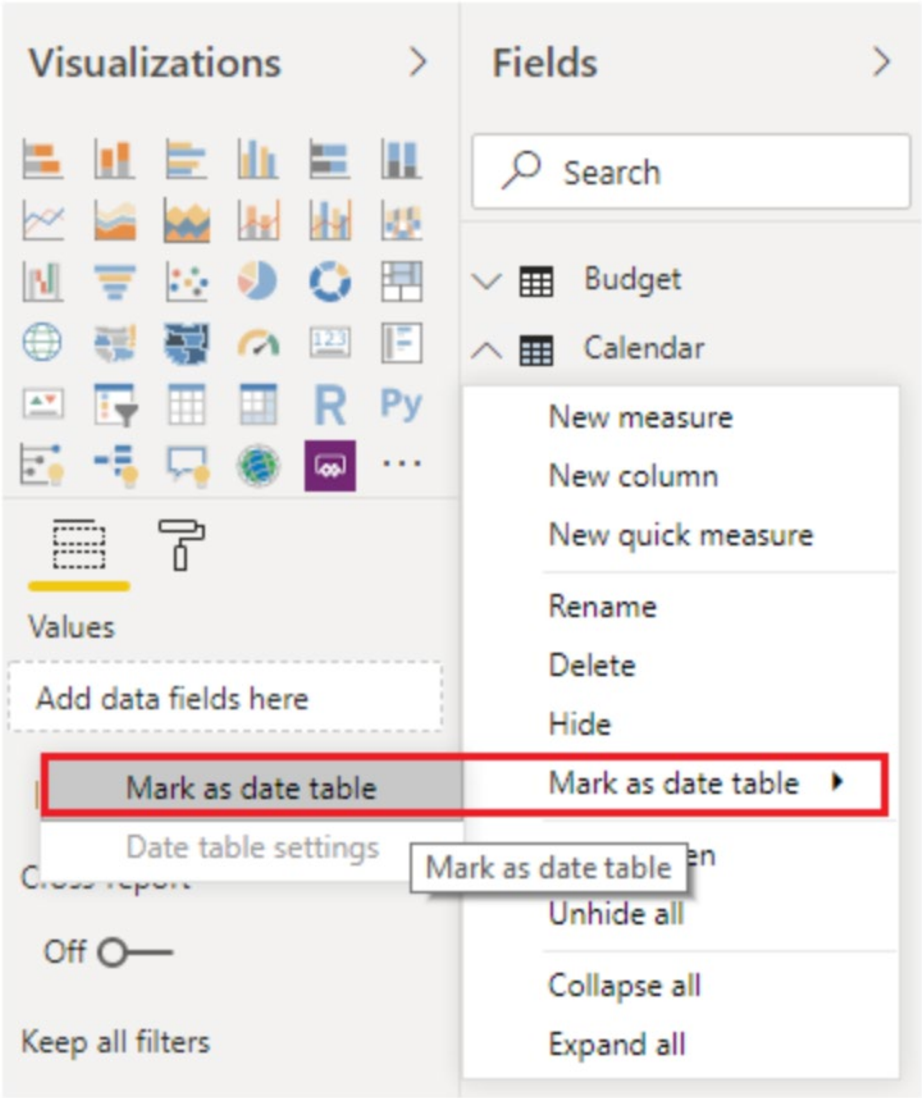
	DateCol	Year	Month	Week of Year	Day Name
1	5/31/2011	2011	5	23	Tuesday
2	6/1/2011	2011	6	23	Wednesday
3	6/2/2011	2011	6	23	Thursday

You have now successfully used Power Query to build a common date table.

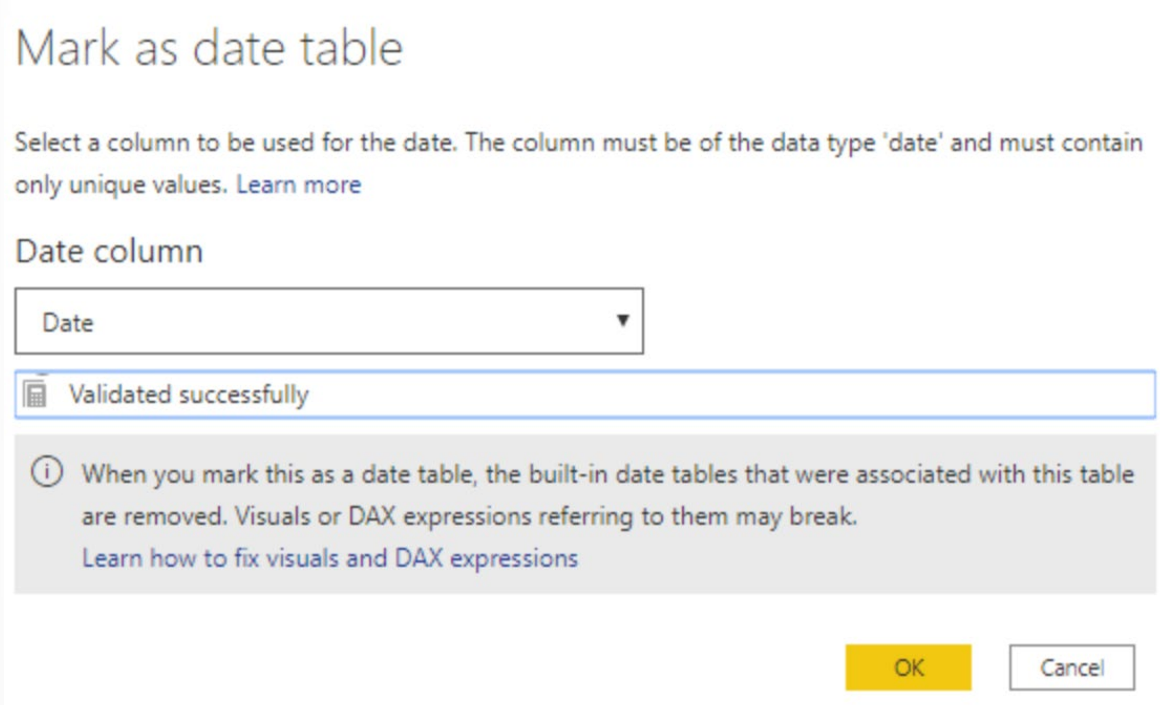
The previous steps show how to get the table into the data model. Now, you need to mark your table as the official date table so that Power BI can recognize it for all future values and ensure that formatting is correct.

Mark as the official date table

Your first task in marking your table as the official date table is to find the new table on the **Fields** pane. Right-click the name of the table and then select **Mark as date table**, as shown in the following figure.



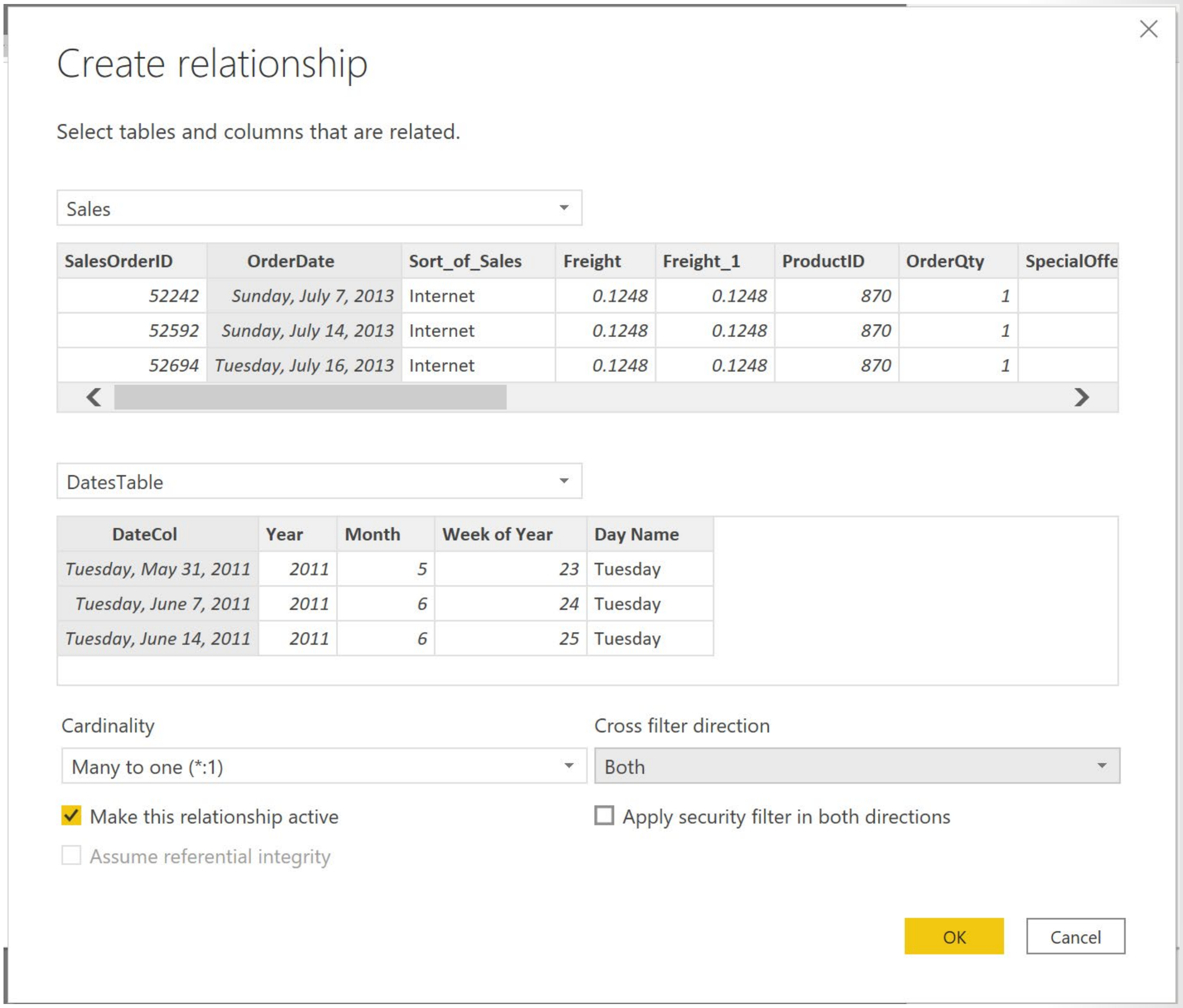
By marking your table as a date table, Power BI performs validations to ensure that the data contains zero null values, is unique, and contains continuous date values over a period. You can also choose specific columns in your table to mark as the date, which can be useful when you have many columns within your table. Right-click the table, select **Mark as date table**, and then select **Date table settings.** The following window will appear, where you can choose which column should be marked as **Date**.



Selecting **Mark as date table** will remove autogenerated hierarchies from the **Date** field in the table that you marked as a date table. For other date fields, the auto hierarchy will still be present until you establish a relationship between that field and the date table or until you turn off the **Auto Date/Time** feature. You can manually add a hierarchy to your common date table by right-clicking the year, month, week, or day columns in the **Fields** pane and then selecting **New hierarchy.** This process is further discussed later in this module.

Build your visual

To build your visual between the Sales and Orders tables, you will need to establish a relationship between this new common date table and the Sales and Orders tables. As a result, you will be able to build visuals by using the new date table. To complete this task, go to **Model** tab **>Manage Relationships**, where you can create relationships between the common date table and the Orders and Sales tables by using the **OrderDate** column. The following screenshot shows an example of one such relationship.



After you have built the relationships, you can build your **Total Sales and Order Quantity by Time** visual with your common date table that you developed by using the DAX or Power Query method.

To determine the total sales, you need to add all sales because the **Amount** column in the Sales table only looks at the revenue for each sale, not the total sales revenue. You can complete this task by using the following measure calculation, which will be explained in later discussions. The calculation that you will use when building this measure is as follows:

```
#Total Sales = SUM(Sales[‘Amount’])
```

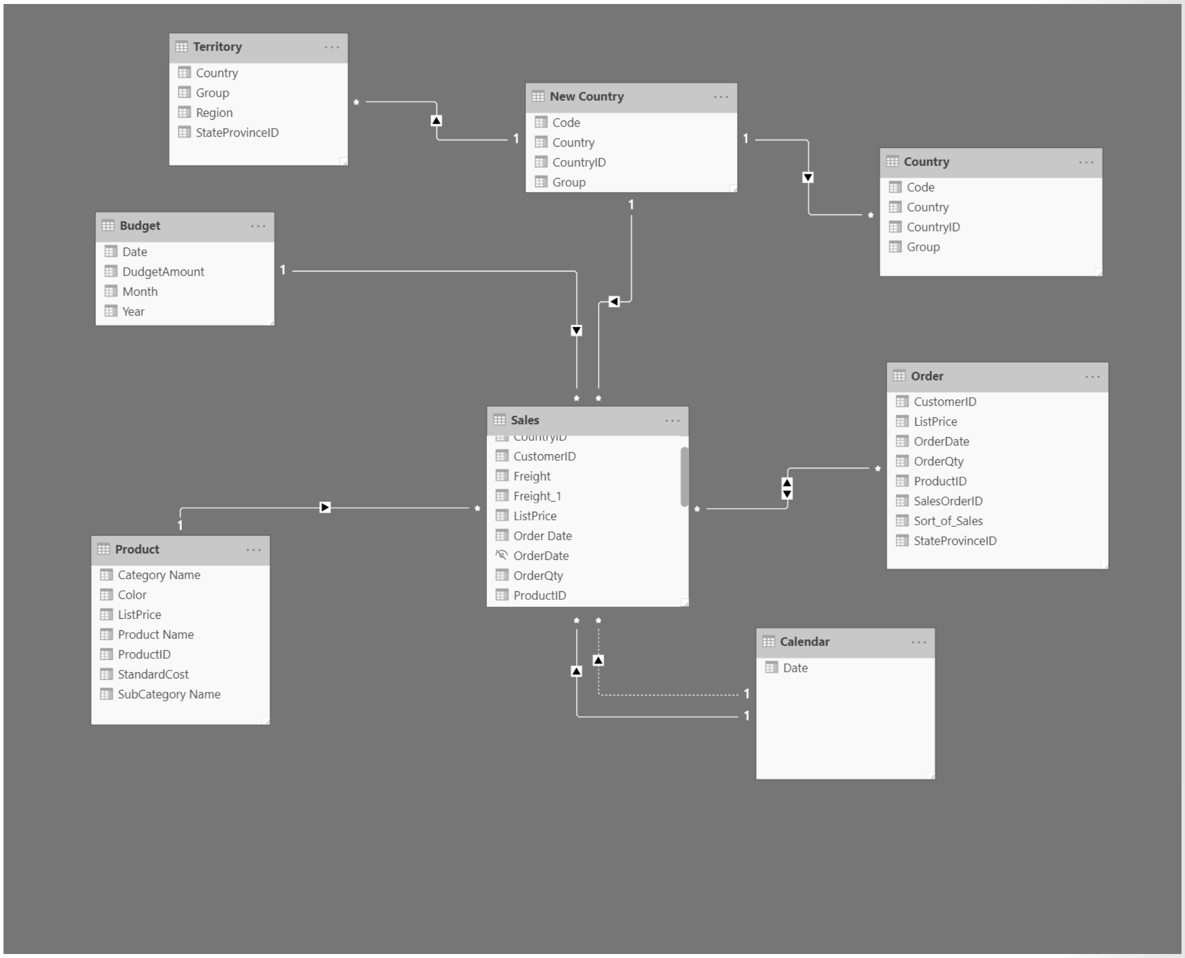
After you have finished, you can create a table by returning to the **Visualizations** tab and selecting the **Table** visual. You want to see the total orders and sales by year and month, so you only want to include the Year and Month columns from your date table, the **OrderQty** column, and the **#TotalSales** measure. When you learn about hierarchies, you can also build a hierarchy that will allow you drill down from years to months. For this example, you can view them side-by-side. You have now successfully created a visual with a common date table.

Year	Month	OrderQty	# TotalSales
2011	5	825	853,422
2011	6	141	460,085
2011	7	2209	3,130,880
2011	8	2904	3,917,345
2011	9	157	503,668
2011	10	5382	7,426,033
2011	11	230	740,105
2011	12	1040	1,815,966
2012	1	3967	6,319,337
2012	2	1442	2,106,429
2012	3	3184	4,575,015
Total		**274914**	**170,964,700**

Relationships and cardinality

Unlike other database management systems, Power BI has the concept of *directionality* to a relationship. This directionality, or *cardinality*, plays an important role in filtering data between multiple tables. When you load data, Power BI automatically looks for relationships that exist within the data by matching column names. You can also use **Manage Relationships** to edit these options manually.

For example, you've retrieved many tables from the Sales database, and the following image is an example of your data model. Power BI has autodetected several relationships, but you can't discern what they mean. You want to make sure that the relationships accurately reflect those that exist in your data.



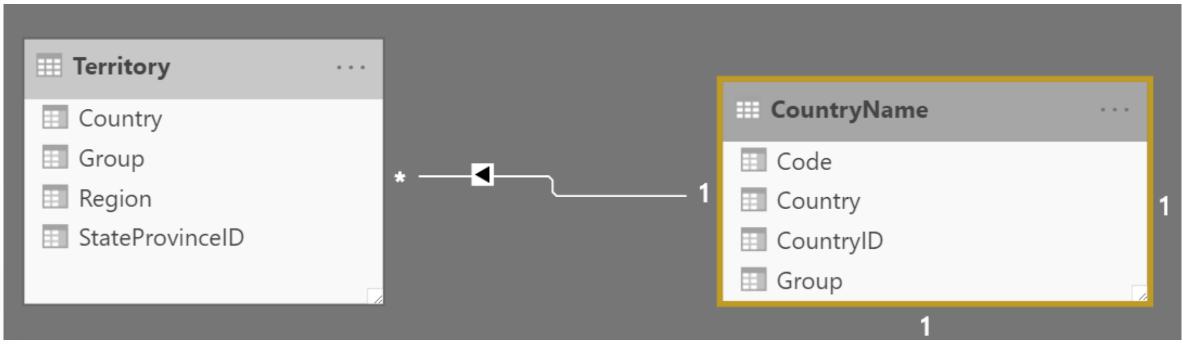
Cardinality

The following are different types of cardinality that you'll find in Power BI.

Many-to-one (*:1) or one-to-many (1: *) cardinality:

- Describes a relationship in which you have many instances of a value in one column that are related to only one unique corresponding instance in another column.
- Describes the directionality between fact and dimension tables.
- Is the most common type of directionality and is the Power BI default when you are automatically creating relationships.

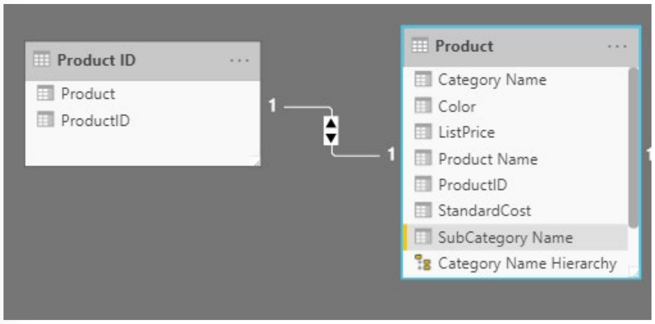
An example of a one-to-many relationship would be between the CountryName and Territory tables, where you can have many territories that are associated with one unique country.



One-to-one (1:1) cardinality:

- Describes a relationship in which only one instance of a value is common between two tables.
- Requires unique values in both tables.
- Is not recommended because this relationship stores redundant information and suggests that the model is not designed correctly. It is better practice to combine the tables.

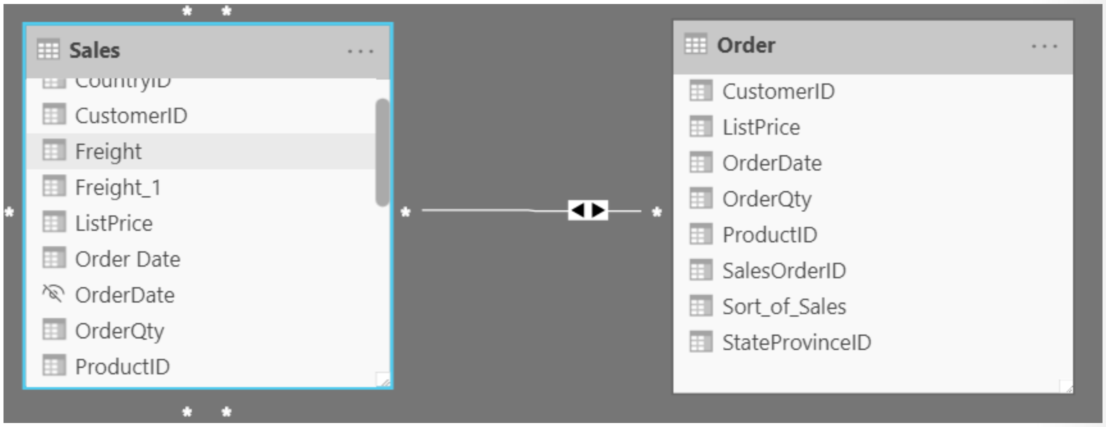
An example of a one-to-one relationship would be if you had products and product IDs in two different tables. Creating a one-to-one relationship is redundant and these two tables should be combined.



Many-to-many (.) cardinality:

- Describes a relationship where many values are in common between two tables.
- Does not require unique values in either table in a relationship.
- Is not recommended; a lack of unique values introduces ambiguity and your users might not know which column of values is referring to what.

For instance, the following figure shows a many-to-many relationship between the Sales and Order tables on the **OrderDate** column because multiple sales can have multiple orders associated with them. Ambiguity is introduced because both tables can have the same order date.



Cross-filter direction

Data can be filtered on one or both sides of a relationship.

With a **single cross-filter direction**:

- Only one table in a relationship can be used to filter the data. For instance, Table 1 can be filtered by Table 2, but Table 2 cannot be filtered by Table 1.

  TIP: Follow the direction of the arrow on the relationship between your tables to know which direction the filter will flow. You typically want these arrows to point to your fact table.

- For a one-to-many or many-to-one relationship, the cross-filter direction will be from the "one" side, meaning that the filtering will occur in the table that has unique values.

With **both cross-filter directions** or **bi-directional cross-filtering**:

- One table in a relationship can be used to filter the other. For instance, a dimension table can be filtered through the fact table, and the fact tables can be filtered through the dimension table.
- You might have lower performance when using bi-directional cross-filtering with many-to-many relationships.

A word of caution regarding bi-directional cross-filtering: You should not enable bi-directional cross-filtering relationships unless you fully understand the ramifications of doing so. Enabling it can lead to ambiguity, over-sampling, unexpected results, and potential performance degradation.

Cardinality and cross-filter direction

For one-to-one relationships, the only option that is available is bi-directional cross-filtering. Data can be filtered on either side of this relationship and result in one distinct, unambiguous value. For instance, you can filter on one Product ID and be returned a single Product, and you can filter on a Product and be returned a single Product ID.

For many-to-many relationships, you can choose to filter in a single direction or in both directions by using bi-directional cross-filtering. The ambiguity that is associated with bi-directional cross-filtering is amplified in a many-to-many relationship because multiple paths will exist between different tables. If you create a measure, calculation, or filter, unintended consequences can occur where your data is being filtered and, depending on which relationship that the Power BI engine chooses when applying the filter, the final result might be different. This situation is also true for bi-directional relationships and why you should be cautious when using them.

For this reason, many-to-many relationships and/or bi-directional relationships are complicated. Unless you are certain what your data looks like when aggregated, these types of open-ended relationships with multiple filtering directions can introduce multiple paths through the data.

Modeling Challenges

Modeling data is about establishing and maintaining relationships so that you can effectively visualize the data in the form that your business requires. When you are creating these relationships, a common pitfall that you might encounter are circular relationships.

For example, you are developing reports for the Sales team and are examining the relationships between tables. In a poorly designed data model, Table 1 has a many-to-one relationship with a column in Table 2, but Table 2 has a one-to-many relationship with Table 3 that has its own relationship with Table 1. This web of relationships is difficult to manage and becomes a daunting task to build visuals because it is no longer clear what relationships exist. Therefore, it is important that you are able to identify circular relationships so that your data is usable.

Relationship dependencies

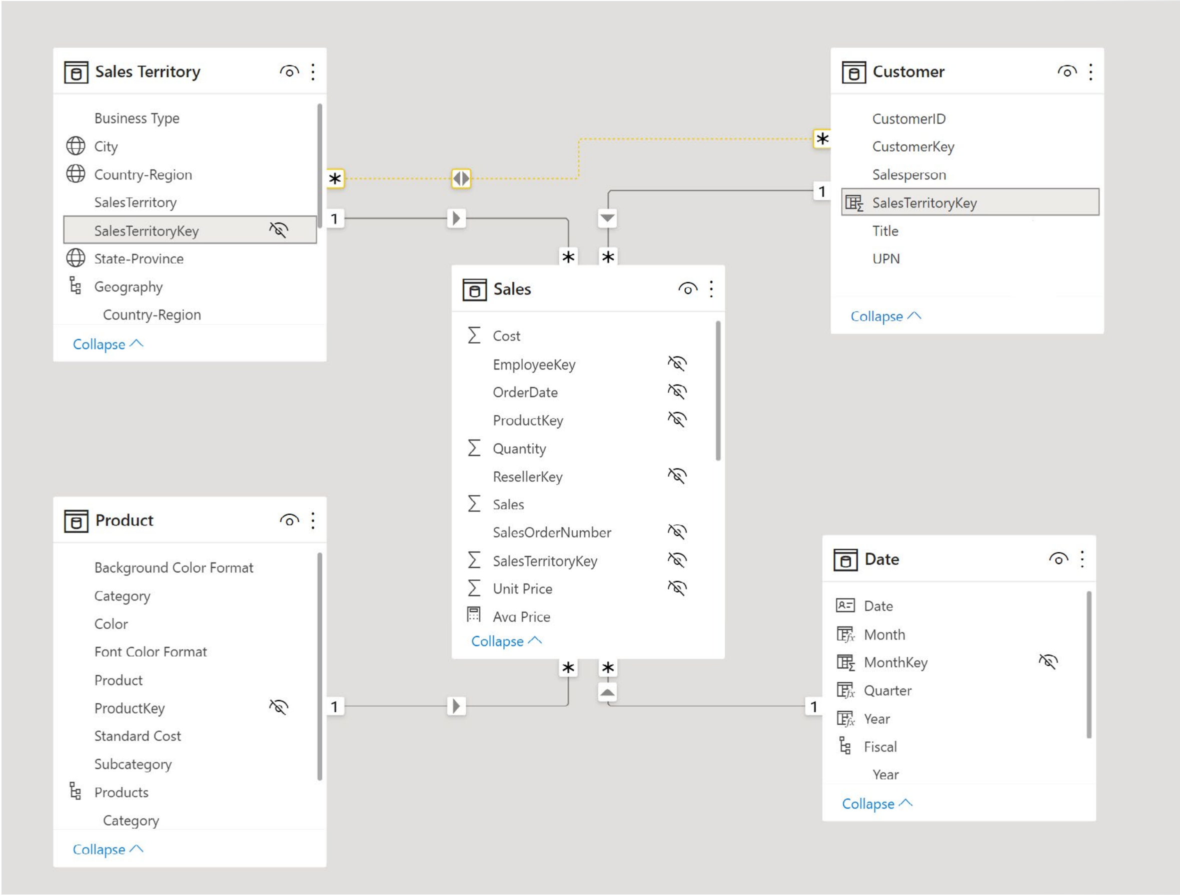
To understand circular relationships, you first need to understand dependencies.

For example, consider that you have the following calculated column **Total** in the Sales table.

Sales['TotalCost] = Sales['Quantity'] * Sales['Price']

TotalCost depends on **Quantity** and **Price**, so if a change occurs in either quantity or price, a change will occur in **TotalCost** as well. This example outlines a dependency of a column on other columns, but you can also have dependencies between measures, tables, and relationships.

Consider the following relationships between **Customer**, **Sales**, and **Sales Territory**. A change in **Customer** will result in a change in **Sales**, which results in changes in **Sale Territory.** These types of dependencies can exist within relationships.



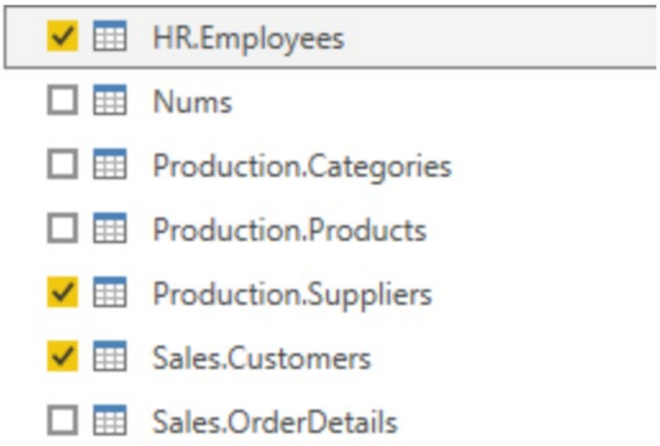
Combine queries

The ability to combine queries is powerful because it allows you to append or merge different tables or queries together. You can combine tables into a single table in the following circumstances:

- Too many tables exist, making it difficult to navigate an overly-complicated data model.
- Several tables have a similar role.
- A table has only a column or two that can fit into a different table.
- You want to use several columns from different tables in a custom column.

You can combine the tables in two different ways: merging and appending.

Assume that you are developing Power BI reports for the Sales and HR teams. They have asked you to create a contact information report that contains the contact information and location of every employee, supplier, and customer. The data is in the HR.Employees, Production.Suppliers, and the Sales.Customers tables, as shown in the following image.



However, this data comes from multiple tables, so the dilemma is determining how you can merge the data in these multiple tables and create one source-of-truth table to create a report from. The inherent functionality of Power BI allows you to combine and merge queries into a single table.

Append queries

When you append queries, you will be adding rows of data to another table or query. For example, you could have two tables, one with 300 rows and another with 100 rows, and when you append queries, you will end up with 400 rows. When you merge queries, you will be adding columns from one table (or query) into another. To merge two tables, you must have a column that is the key between the two tables.

For the previously mentioned scenario, you will append the HR.Employees table with the Production.Suppliers and Sales.Customers tables so that you have one master list of contact information. Because you want to create one table that has all contact information for employees, suppliers, and customers, when you combine the queries, the pertinent columns that you require in your combined table must be named the same in your original data tables to see one consolidated view.

Before you begin combining queries, you can remove extraneous columns that you don't need for this task from your tables. To complete this task, format each table to have only four columns with your pertinent information, and rename them so they all have the same column headers: ID, company, name, and phone. The following images are snippets of the reformatted Sales.Customers, Production.Suppliers, and HR.Employees tables.

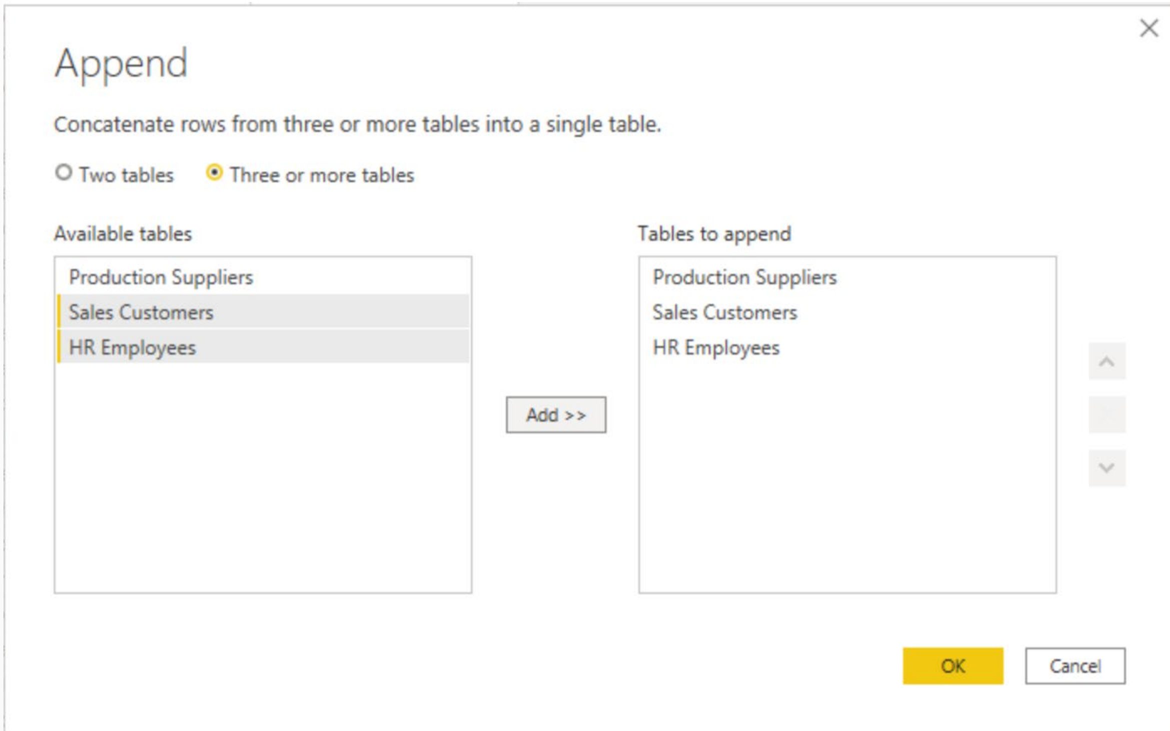
	id	company	name	phone
1	96351	cus145	Mike Poi	1626258811
2	64826	cus134	John Likec	1363718361
3	92647	cus018	Theresa Yulia	2467355553

	id	company	name	phone
1	86563	sup126	Mike Firt	7163615121
2	96352	sup889	Josie Lind	1831635671
3	48256	sup761	Joanna Threven	2936827338

	id	company	name	phone
1	12347	emp124	John Kate	1625535511
2	76478	emp273	Luke John	1028777351
3	82482	emp291	Michael Uji	1245245672

After you have finished reformatting, you can combine the queries. On the **Home** tab on the Power Query Editor ribbon, select the drop-down list for **Append Queries**. You can select **Append Queries as New**, which means that the output of appending will result in a new query or table, or you can select **Append Queries**, which will add the rows from an existing table into another.

Your next task is to create a new master table, so you need to select **Append Queries as New**. This selection will bring you to a window where you can add the tables that you want to append from **Available Tables** to **Tables to Append**, as shown in the following image.



After you have added the tables that you want to append, select **OK**. You will be routed to a new query that contains all rows from all three of your tables, as shown in the following image.

	id	company	name	phone
1	12347	emp124	John Kate	1625535511
2	76478	emp273	Luke John	1028777351
3	82482	emp291	Michael Uji	1245245672
4	97436	emp173	Kate Fitch	2352467634
5	12462	emp270	Eve Jun	3578999554
6	35237	emp715	Don Potre	3579006677
7	23467	emp183	Marc Wetb	2245789954
8	13892	emp163	Sara Scotts	2388367234
9	56356	emp172	Mitch Potter	1234683673
10	23478	emp812	Liliy Kithc	4567800522
11	45783	emp818	Ren Swrte	2357997515
12	86563	sup126	Mike Firt	7163615121
13	96352	sup889	Josie Lind	1831635671
14	48256	sup761	Joanna Threven	2936827338
15	28461	sup163	Michael Bob	1937293165
16	83613	sup162	Mimi Jukth	2916384462
17	96351	cus145	Mike Poi	1626258811
18	64826	cus134	John Likec	1363718361
19	92647	cus018	Theresa Yulia	2467355553
20	91661	cus182	Ren Thibe	3345783234
21	1736	cus104	Ron Mikel	1235799789
22	1835	cus103	Joy Qui	2345689411
23	1745	cus141	Cat Yate	2345678986

You have now succeeded in creating a master table that contains the information for the employees, suppliers, and customers. You can exit Power Query Editor and build any report elements surrounding this master table.

However, if you wanted to merge tables instead of appending the data from one table to another, the process would be different.

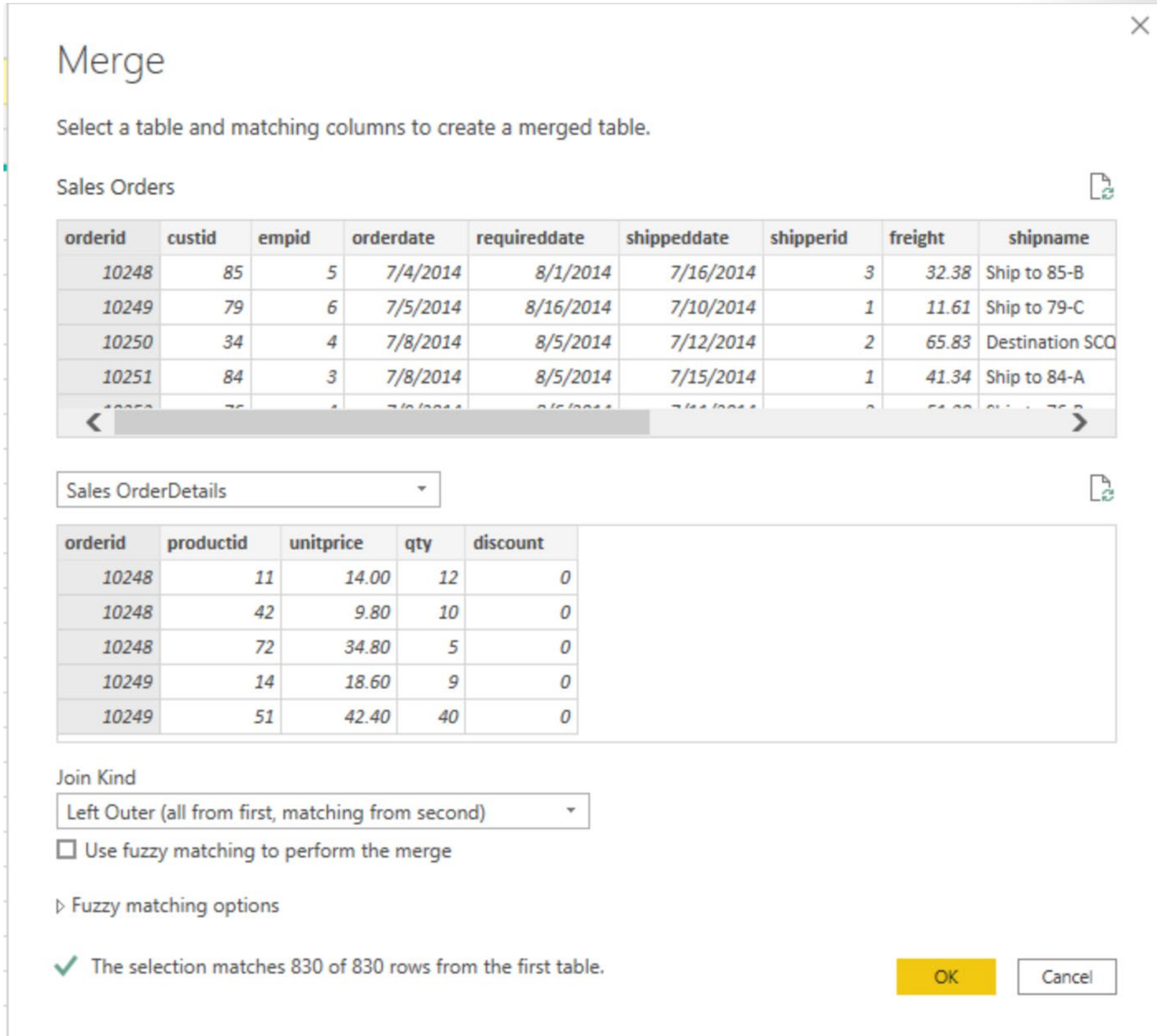
Merge queries

When you merge queries, you are combining the data from multiple tables into one based on a column that is common between the tables. This process is similar to the JOIN clause in SQL. Consider a scenario where the Sales team now wants you to consolidate orders and their corresponding details (which are currently in two tables) into a single table. You can accomplish this task by merging the two tables, Orders and OrderDetails, as shown in the following image. The column that is shared between these two tables is **OrderID**.

	orderid	orderdate	shipperid
1	1	4/23/2018	12
2	2	4/25/2018	24
3	3	6/12/2018	19
4	4	6/13/2018	13
5	5	7/23/2018	11
6	6	7/25/2018	33

	orderid	productid	qty	unitprice
1	1	124	12	14
2	2	134	55	11.2
3	3	641	57	45
4	4	98	5	112.5
5	5	312	23	11.1
6	6	124	78	11.2

Go to **Home** on the Power Query Editor ribbon and select the **Merge Queries** drop-down menu, where you can select **Merge Queries as New**. This selection will open a new window, where you can choose the tables that you want to merge from the drop-down list, and then select the column that is matching between the tables, which in this case is **orderid**.



You can also choose how to join the two tables together, a process that is also similar to JOIN statements in SQL. These join options include:

- **Left Outer** - Displays all rows from the first table and only the matching rows from the second.
- **Full Outer** - Displays all rows from both tables.
- **Inner** - Displays the matched rows between the two tables.

For this scenario, you will choose to use a **Left Outer** join. Select **OK**, which will route you to a new window where you can view your merged query.

	orderid	orderdate	shipperid	OrderDetails.productid	OrderDetails.qty	OrderDetails.unitprice
1	1	4/23/2018	12	124	12	14
2	2	4/25/2018	24	134	55	11.2
3	3	6/12/2018	19	641	57	45
4	4	6/13/2018	13	98	5	112.5
5	5	7/23/2018	11	312	23	11.1
6	6	7/25/2018	33	124	78	11.2
7	7	8/1/2019	77	137	11	572.1
8	8	8/10/2019	11	124	36	1331.9
9	9	8/11/2019	81	789	85	898.1

Now, you can merge two queries or tables in different ways so that you can view your data in the most appropriate way for your business requirements.

For more information on this topic, see the **Shape and Combine Data in Power BI**[1] documentation.

Knowledge Check

Question 1

What is Cardinality?

- ☐ Cardinality is how long it takes for the data to load
- ☐ Cardinality is the granularity of the data
- ☐ The direction that the data flows in a relationship between two tables
- ☐ Cardinality is a type of visual element

1 https://docs.microsoft.com/power-bi/connect-data/desktop-shape-and-combine-data

Dimensions and Hierarchies

Introduction to Dimensions and Hierarchies

When building a star schema, you will have dimension and fact tables. Fact tables contain information about events such as sales orders, shipping dates, resellers, and suppliers. Dimension tables store details about business entities, such as products or time, and are connected back to fact tables through a relationship.

You can use hierarchies as one source to help you find detail in dimension tables. These hierarchies form through natural segments in your data. For instance, you can have a hierarchy of dates in which your dates can be segmented into years, months, weeks, and days. Hierarchies are useful because they allow you to drill down into the specifics of your data instead of only seeing the data at a high level.

Hierarchies

When you are building visuals, Power BI automatically enters values of the date type as a hierarchy (if the table has not been marked as a date table).

In the preceding **Date** column, the date is shown in increasingly finer detail through year, quarters, months, and days. You can also manually create hierarchies.

Parent-child hierarchy

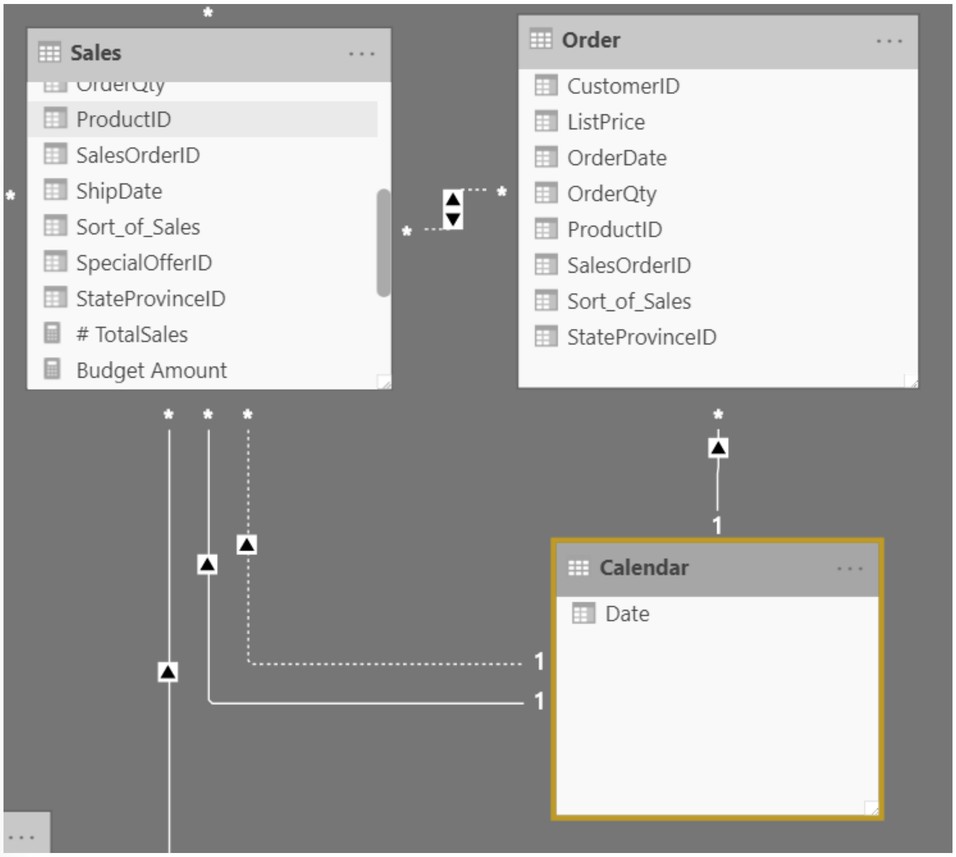
In the following example, you have an Employee table within the database that tells you important information about the employees, their managers, and their IDs. When looking at this table, you notice that **Roy F** has been repeated multiple times in the **Manager** column. As the image shows, multiple employees can have the same manager, which indicates a hierarchy between managers and employees.

	Employee ID	Employee	Manager ID	Manager
1	1010	Roy F	null	
2	1011	Pam H	1010	Roy F
3	1012	Guy L	1010	Roy F
4	1013	Roger M	1011	Pam H
5	1014	Kaylie S	1011	Pam H
6	1015	Mike O	1012	Guy L
7	1016	Rudy Q	1012	Guy L

The **Manager** column determines the hierarchy and is therefore the parent, while the “children” are the employees. For this example, you want to be able to see all levels of this hierarchy. Power BI does not default to showing you all levels of the hierarchy, so it is your responsibility to ensure that you see all levels of this hierarchy or "flatten" it so that you can see more data granularity.

Role-playing dimensions

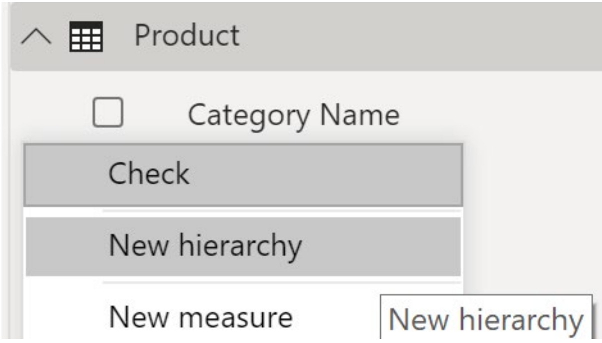
Role-playing dimensions have multiple valid relationships with fact tables, meaning that the same dimension can be used to filter multiple columns or tables of data. As a result, you can filter data differently depending on what information you need to retrieve. This topic is complex, so it is only introduced in this section. Working with role-playing dimensions requires complex DAX functions that will be discussed in later sections.



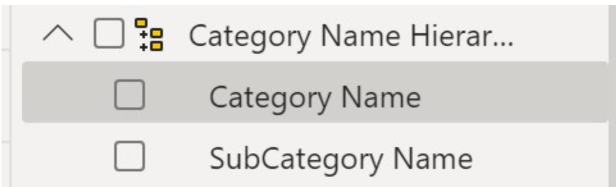
The preceding visual shows the Calendar, Sales, and Order tables. Calendar is the dimension table, while Sales and Order are fact tables. The dimension table has two relationships: one with Sales and one with Order. This example is of a role-playing dimension because the Calendar table can be used to group data in both Sales and Order. If you wanted to build a visual in which the Calendar table references the Order and the Sales tables, the Calendar table would act as a role-playing dimension.

Creating new Hierarchies

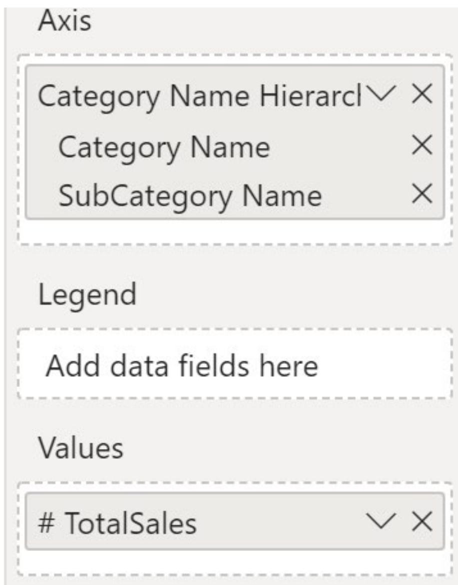
Consider a situation where you want to create a stacked bar chart of **Total Sales by Category and Subcategory**. You can accomplish this task by creating a hierarchy in the **Product** table for categories and subcategories. To create a hierarchy, go to the **Fields** pane on Power BI and then right-click the column that you want the hierarchy for. Select **New hierarchy**, as shown in the following figure.



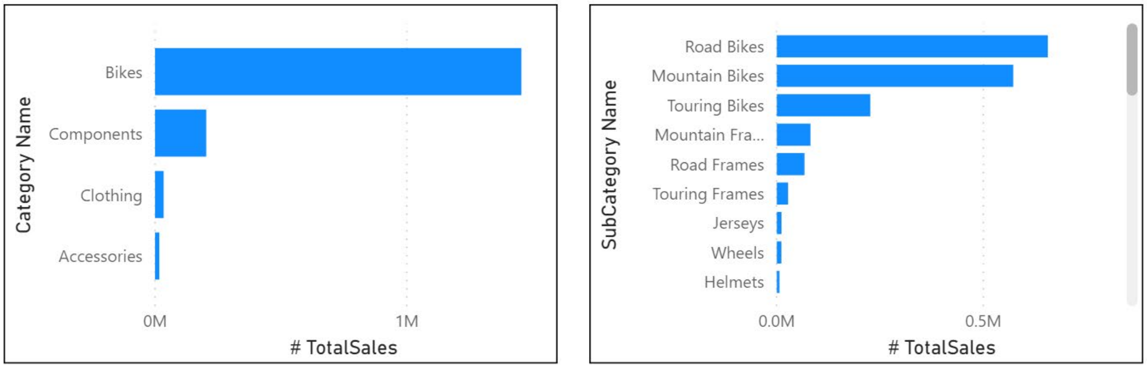
Next, drag and drop the subcategory column into this new hierarchy that you've created. This column will be added as a sublevel on the hierarchy.



Now, you can build the visual by selecting a stacked bar chart in the **Visualizations** pane. Add your **Category Name Hierarchy** in the **Axis** field and the **Total Sales** hierarchy in the **Values** field.



You can drill down on the visual to view both **Category** and **Subcategory**, depending on what you want to see. Hierarchies allow you to view increasing levels of data on a single view.



Module Review

You have learned about modeling data in Power BI, which includes such topics as creating common date tables, resolving circular relationships, designing star schemas, and much more. These skills are crucial to the Power BI practitioner's toolkit so that it is easier to build visuals and hand off your report elements to other teams. With this foundation, you now have the ability to explore the many nuances of the data model.

Knowledge Check

Question 1

A dimension that can filter related facts differently is called what?

☐ Role-playing dimension

☐ Snowflake dimension

☐ Degenerate dimension

Question 2

What type of table stores details about business entities?

☐ Fact table

☐ Dimension table

☐ Date table

☐ Data table

Answers

Question 1

The two types of tables in a star schema are what?

- ☐ Active and inactive tables
- ☐ Qualitative and quantitative data tables
- ■ Fact and dimension tables

Question 2

What is the difference between a fact table and a dimension table?

- ■ Fact tables store observations or events while dimension tables contain information about specific entities within the data
- ☐ Fact tables contain information about specific entities while dimension tables contain information about observational data
- ☐ Dimension tables tell you about specific roles in Power BI while fact tables tell you information about facts that are associated with those roles in Power BI
- ☐ There is no difference

Question 1

What is Cardinality?

- ☐ Cardinality is how long it takes for the data to load
- ☐ Cardinality is the granularity of the data
- ■ The direction that the data flows in a relationship between two tables
- ☐ Cardinality is a type of visual element

Question 1

A dimension that can filter related facts differently is called what?

- ■ Role-playing dimension
- ☐ Snowflake dimension
- ☐ Degenerate dimension

Question 2

What type of table stores details about business entities?

☐ Fact table

■ Dimension table

☐ Date table

☐ Data table

Module 5 Create Model Calculations using DAX in Power BI

Introduction to DAX

What is DAX

Data Analysis Expressions (DAX) is a programming language that is used throughout Microsoft Power BI for creating calculated columns, measures, and custom tables. It is a collection of functions, operators, and constants that can be used in a formula, or expression, to calculate and return one or more values. You can use DAX to solve a number of calculations and data analysis problems, which can help you create new information from data that is already in your model.

In Power BI, you can use different calculation techniques and functions to create measures or calculated columns. Primarily, you will be able to achieve the same result by using these techniques; however, the key is to know how and when to apply them. By having a basic understanding of when and how to use which technique, you will be able to create robust and high-performance data models.

By the end of this module, you'll be able to:

- Build quick measures.
- Create calculated columns.
- Use DAX to build measures.
- Discover how context affects DAX measures.
- Use the CALCULATE function to manipulate filters.
- Implement time intelligence by using DAX.

Measures

Use measures

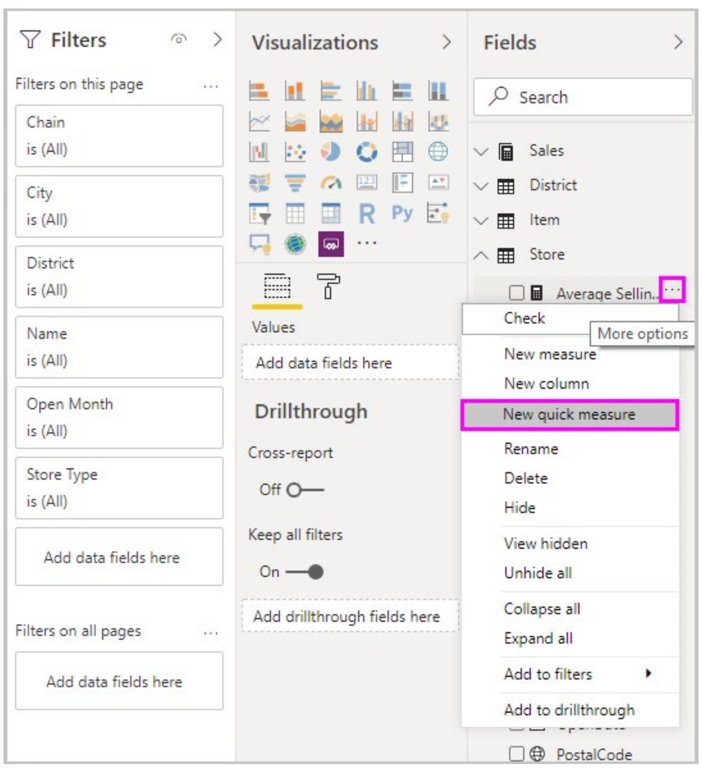
Calculated columns are useful, but you are required to operate row by row. However, other situations might require a simpler method. For example, consider a situation where you want an aggregation that operates over the entire dataset and you want the total sales of all rows. Furthermore, you want to slice and dice that data by other criteria like total sales by year, by employee, or by product.

To accomplish those tasks, you would use a measure. You can build a measure without writing DAX code; Power BI will write it for you when you create a quick measure.

Many available categories of calculations and ways to modify each calculation exist to fit your needs. Another advantage is that you can see the DAX that's implemented by the quick measure while jump-starting or expanding your own DAX knowledge.

Create a quick measure

To create a quick measure in Power BI Desktop, right-click or select the ellipsis (**...**) button next to any item in the **Fields** pane and then select **New quick measure** from the menu that appears. The **Quick measures** screen will appear.



In the **Quick measures** window, you can select the calculation that you want and the fields to run the calculation against. For instance, you can select a calculation and the column that you want to operate over. Power BI creates the DAX measure for you and displays the DAX. This approach can be a helpful way to learn the DAX syntax.

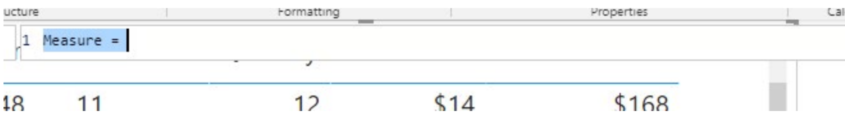
For more information, see the **Use quick measures for common calculations**[1] documentation.

Create a measure

Measures are used in some of the most common data analyses.

To continue with the previous scenario, you want to create a measure that totals your new column for the entire dataset. Similar to how you created a calculated column, you can go to the **Fields** list and select **New measure**.

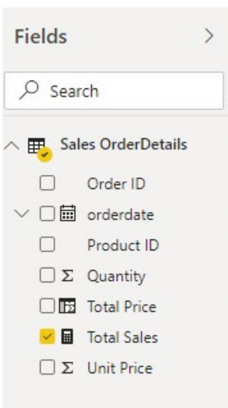
Text will now appear in the formula bar underneath the ribbon.



You can replace the "Measure =" text with the following text:

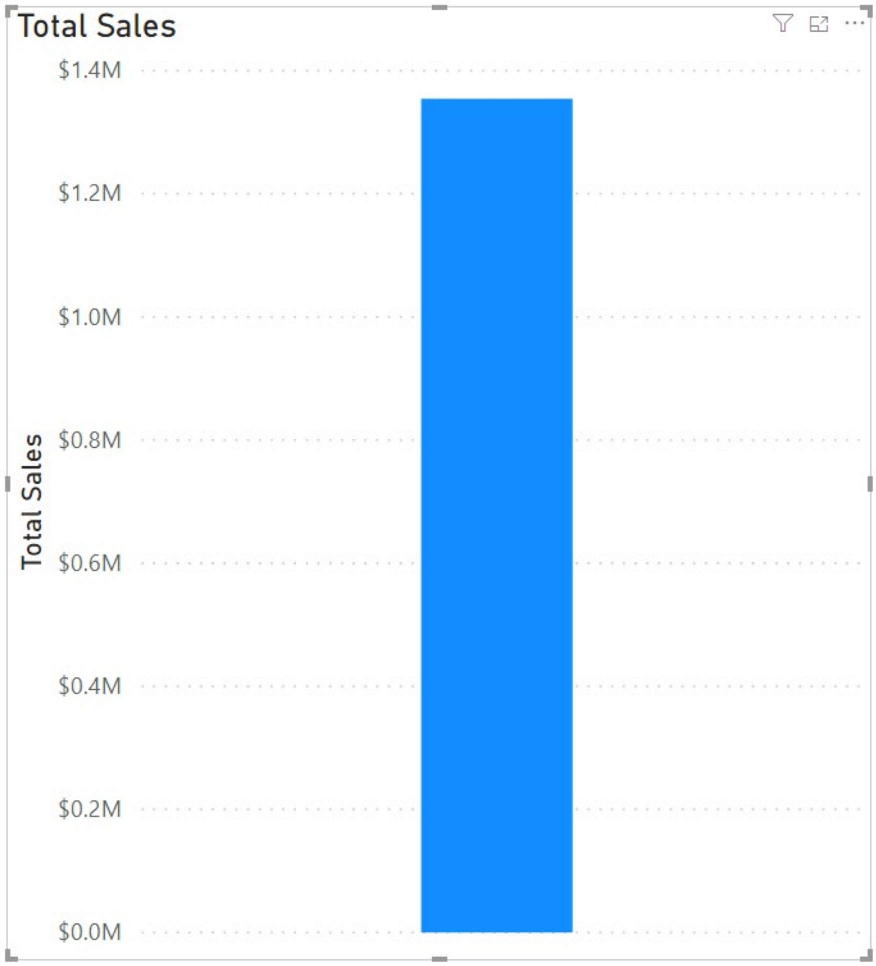
```
Total Sales = sum('Sales OrderDetails'[Total Price])
```

The new measure will now appear in the **Fields** list.



When you drag Total Sales over to the report design surface, you will see the total sales for the entire organization in a column chart.

1 https://docs.microsoft.com/power-bi/desktop-quick-measures/



Calculated columns

Use calculated columns

DAX allows you to augment the data that you bring in from different data sources by creating a calculated column that didn't originally exist in the data source. This feature should be used sparingly, which will be explained later in this module.

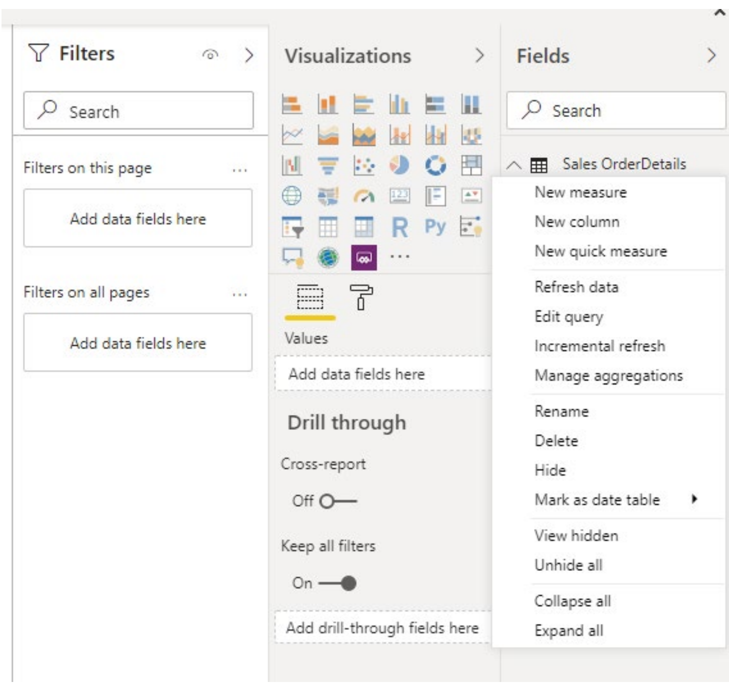
For example, assume that you are importing data from a database that contains sales transactions. Each individual sales transaction has the following columns: **Order ID**, **Product ID**, **Quantity**, and **Unit Price**. Notice that a column doesn't exist for the total sales amount for each order.

NOTE: This module is not about data visualization, but it does show data visualization to demonstrate how DAX works. For more information, see the learning path, **Visualize data in Power BI**.

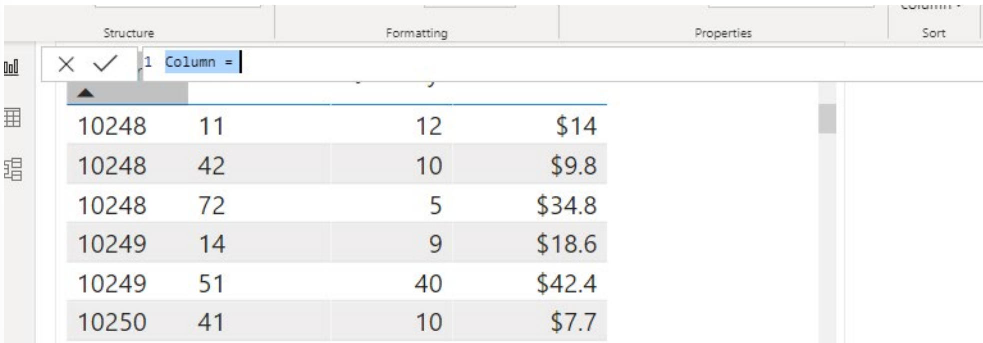
The following figure shows how the initial shape of the data appears in a Power BI table visual.

Order ID	Product ID	Quantity	Unit Price
10248	11	12	$14
10248	42	10	$9.8
10248	72	5	$34.8
10249	14	9	$18.6
10249	51	40	$42.4
10250	41	10	$7.7
10250	51	35	$42.4
10250	65	15	$16.8
10251	22	6	$16.8
10251	57	15	$15.6
10251	65	20	$16.8
10252	20	40	$64.8
10252	33	25	$2
10252	60	40	$27.2
10253	31	20	$10

You can start using DAX by creating a calculated column that multiplies the unit price with the quantity. The calculated column will create a value for each row called Total Price. Create the new column by selecting the ellipsis (**...**) button on the table in the **Fields** list and then selecting **New column**.



A new DAX formula appears in the formula bar underneath the ribbon at the top.



You can replace the "Column =" default text with the following example text:

```
Total Price = 'Sales OrderDetails'[Quantity] * 'Sales OrderDetails'[Unit
Price]
```

The value on the left side of the equal sign is the column name. The text on the right side of the equal sign is the DAX expression. This simple DAX expression takes the quantity value and multiplies it with the unit price value for each individual row. It will produce one value for each record in the table. If you drag the new column from the **Fields** list to the visual, you will see the new values.

Order ID	Product ID	Quantity	Unit Price	Total Price
10248	11	12	$14	$168
10248	42	10	$9.8	$98
10248	72	5	$34.8	$174
10249	14	9	$18.6	$167.4
10249	51	40	$42.4	$1,696
10250	41	10	$7.7	$77
10250	51	35	$42.4	$1,484
10250	65	15	$16.8	$252
10251	22	6	$16.8	$100.8
10251	57	15	$15.6	$234
10251	65	20	$16.8	$336
10252	20	40	$64.8	$2,592
10252	33	25	$2	$50
10252	60	40	$27.2	$1,088

The previous screenshot shows that DAX is calculating correctly and displaying the results that you wanted.

Calculated columns are materialized in the .pbix Power BI file extension, meaning that each time you add a calculated column, you are increasing the size of the overall file. Having too many calculated columns will slow performance and will cause you to reach the maximum Power BI data size sooner.

Create a custom column

Three ways to create a custom column in Power BI are:

- Create the column in the source query when you get the data, for instance, by adding the calculation to a view in a relational database.

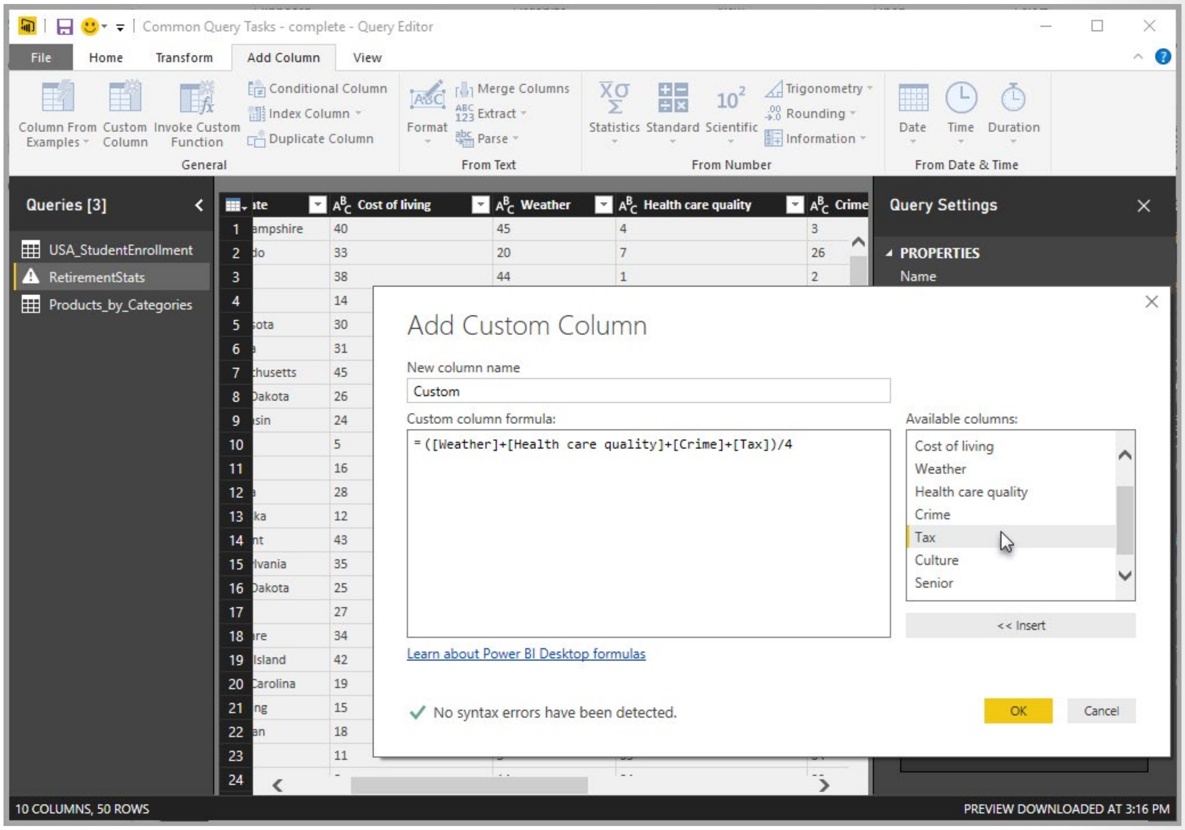
- Create the custom column in Power Query.
- Create a calculated column by using DAX in Power BI.

You can create a calculated column when you pull the data from the data source. Each data source would have a different technique for completing this action. For instance, if you were pulling data from a relational data source by using a view that was written in the SQL language, it would look like the following example:

```
CREATE VIEW OrdersWithTotalPrice
AS
SELECT unitprice, qty, unitprice * qty as TotalPrice
FROM sales.salesorders
```

Using SQL language is an efficient way of creating a column because it would make the data source do the calculations for you. In Power BI, the calculated column would appear like any other column.

You can also use Power Query to create a custom column.



The custom column dialog uses the M language to create the new column. M language is out of scope for the purposes of this module.

The third way to create a calculated column is by using DAX in Power BI, as previously demonstrated.

When you create a calculated column by using DAX, you do not need to refresh the dataset to see the new column. In the other methods, you would need a refresh to see changes. This process can be lengthy if you are working with a lot of data. However, this issue is irrelevant because, after columns have been created, they are rarely changed.

The DAX calculated column does not compress as well as the other methods. The other column types do get compressed, which makes the .pbix file smaller and the performance usually faster.

Generally, the earlier you can create a column, the better. It is not considered an optimum practice to use DAX for calculations if you can use a different mechanism.

In addition, one way to avoid using a calculated column is to use one of the X functions, such as SUMX, COUNTX, MINX, and so on. The X functions are beyond the scope of this module; however, they allow you to create measures that are aware of the data in individual rows and calculate totals based on the totals in the row. These functions are called iterator functions because, though they are used in measures, they iterate over the individual rows to do their calculations. An X function will perform better and use less disk space than a calculated column. For more information about X functions, see the **Microsoft documentation**[2].

Columns vs measures

Differences between a calculated column and a measure

The fundamental difference between a calculated column and a measure is that a calculated column creates a value for each row in a table. For this reason, the calculated column can only operate over columns that exist in the same table. For example, if the table has 1,000 rows, it will have 1,000 values in the calculated column. Calculated column values are stored in the Power BI .pbix file. Each calculated column will increase the space that is used in that file and potentially increase the refresh time.

Measures are calculated on demand. Power BI calculates the correct value when the user requests it. When you previously dragged the Total Sales measure onto the report, Power BI calculated the correct total and displayed the visual. Measures do not add to the overall disk space of the Power BI .pbix file.

Measures are calculated based on the filters that are used by the report user. These filters combine to create the filter context.

Knowledge Check

Question 1

Which are calculated on demand?

☐ Calculated columns

☐ Calculated tables

☐ Measures

Question 2

Which are calculated based on the filters that are used by the report user? Calculated columns or measures?

☐ Measures

☐ Calculated columns

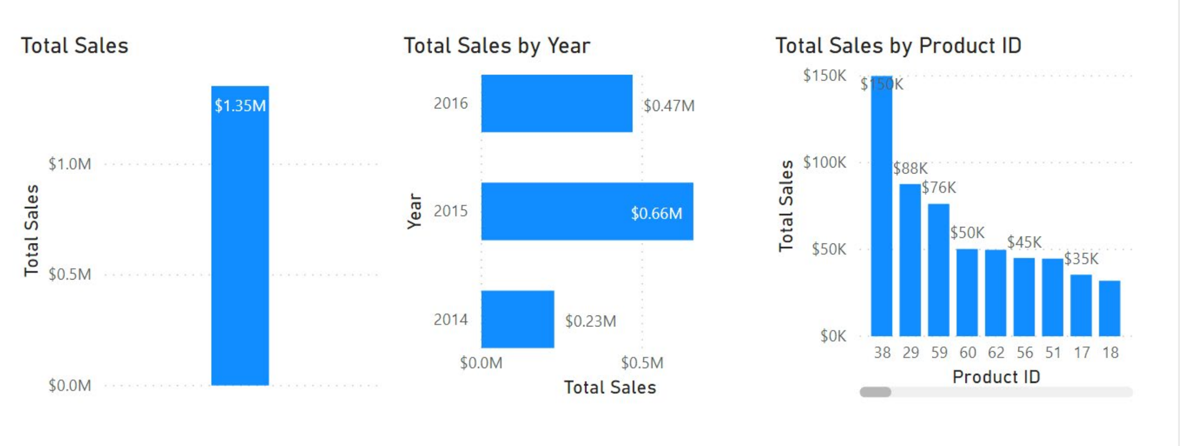
2 https://docs.microsoft.com/dax/sumx-function-dax/

Real-time Dashboards

Understanding Context

How context affects DAX measures is a difficult concept to comprehend. The ensuing visuals will demonstrate how context affects DAX measures so you can see how they interact together.

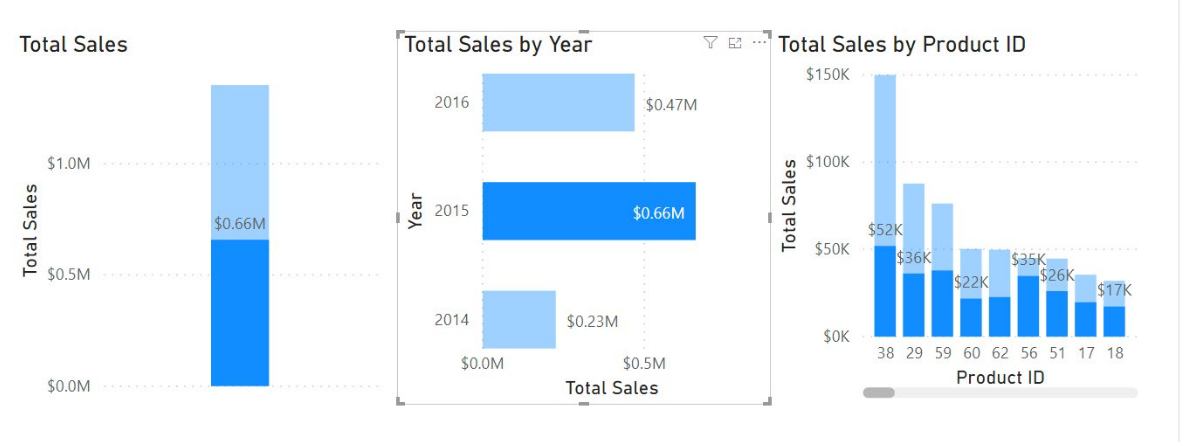
The following three visuals use the exact same DAX measure: Total Sales.



Though each visual uses the same DAX measure and, therefore, the same DAX formula, the visuals produce different results. For instance, the first visual shows the Total Sales measure for the entire dataset. In this dataset, Total Sales is USD1.35 million. In the second visual, Total Sales is broken down by year. For instance, in 2014, Total Sales is USD0.23 million. In the third visual, Total Sales is broken down by Product ID.

With Power BI, even though the measure was only defined once, it can be used in these visuals in different ways. Each of the totals is accurate and performs quickly. It is the context of how the DAX measure is used that calculates these totals accurately.

Interactions between visuals will also change how the DAX measure is calculated. For instance, if you select the second visual and then select **2015**, the results appear as shown in the following screenshot.



Selecting **2015** in the second visual changed the filter context for the DAX measure. It modified the first visual to equal the sales for 2015: USD0.66 million. It also broke down the Total Sales By Product ID, but only shows the results for 2015. Those calculations quickly changed in memory and displayed the results in a highly interactive manner to the user.

The definition of the DAX measure has not changed; it's still the original, as shown in the following example:

```
Total Sales = sum('Sales OrderDetails'[Total Price])
```

This scenario is a simple way to explain how context works with DAX. Many other factors affect how DAX formulas are evaluated. Slicers, page filters, and more can affect how a DAX formula is calculated and displayed.

The CALCULATE function

The CALCULATE function in DAX is one of the most important functions that a data analyst can learn. The function name does not adequately describe what it is intended to do.

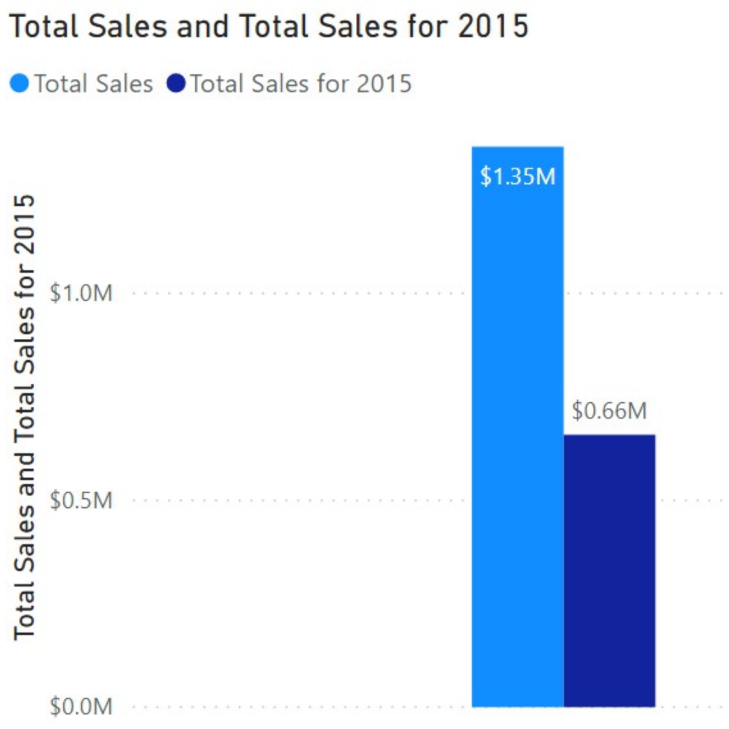
The CALCULATE function is your method of creating a DAX measure that will override certain portions of the context that are being used to express the correct result.

For instance, if you want to create a measure that always calculates the total sales for 2015, regardless of which year is selected in any other visual in Power BI, you would create a measure that looks like the following sample:

```
Total Sales for 2015 = CALCULATE(SUM('Sales OrderDetails'[Total Price]),
YEAR('Sales OrderDetails'[orderdate]) = 2015)
```

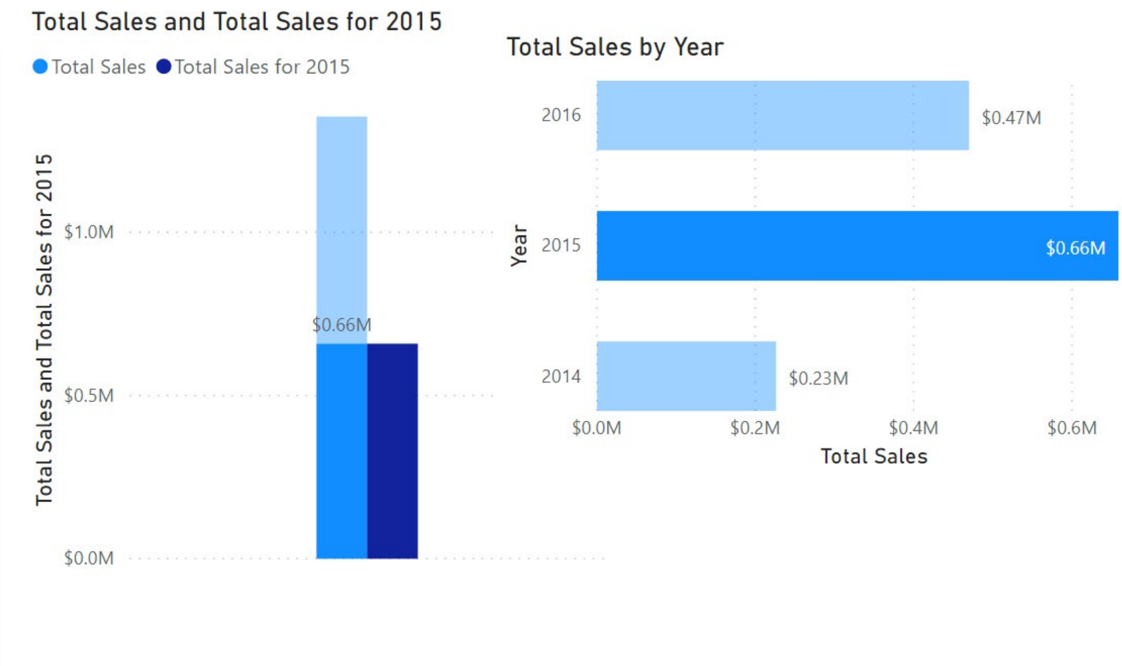
Notice how the measure is named **Total Sales for 2015**. When you use the CALCULATE function to override the context, it is helpful to name the measure in a way that describes exactly how you are overriding it. In this example, CALCULATE is aggregating the Total Price column, just as you did in the previous measure. However, instead of operating over the entire dataset while using whatever the filter context tells it to do, you are overriding the filter context for the year 2015. No matter what year is selected, you will always get the total for 2015; all other filters still apply. The subsequent example shows this concept in action.

When both measures are added to the previous visual they will resemble the following screenshot.



As shown in the preceding screenshot, Total Sales is still USD1.35 million, while the 2015 Total Sales is USD0.66 million.

When you add the other visual onto the report, as you did previously, and then select 2015, the results will look like the following image.



Notice how both measures are now equally the same amount. If you were to filter by any other criteria, including region, employee, or product, the filter context would still be applied to both measures. It's only the year filter that does not apply to that measure.

Knowledge Check

Question 1

Which DAX function evaluates an expression in a modified filter context?

☐ SUMX

☐ CALCULATE

☐ ALL

Question 2

Why would you want to override the default context?

☐ To create measures that behave according to the user's selection

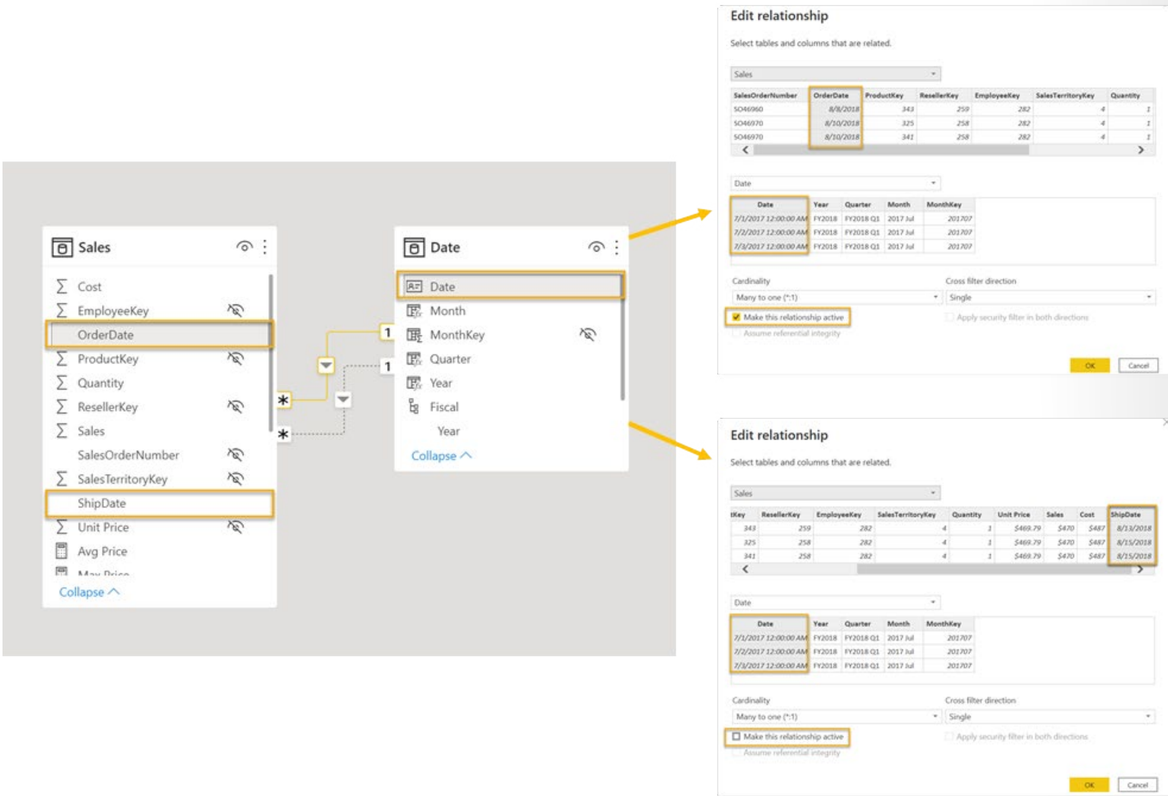
☐ To create measures that behave according to your intentions, regardless of what the user selects

Advanced DAX

Using relationships effectively

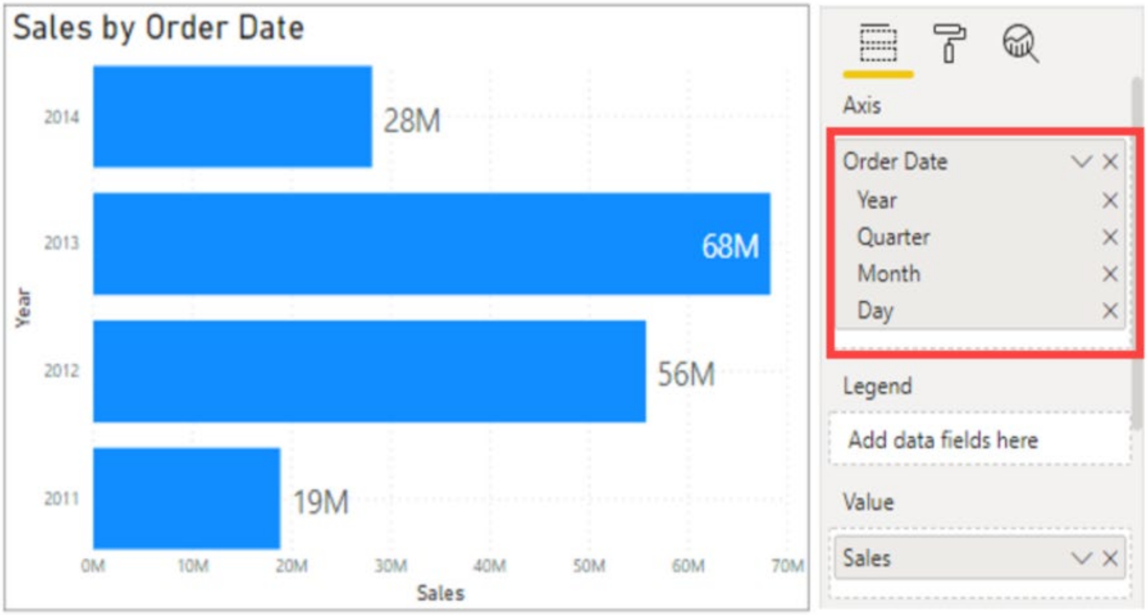
Another DAX function that allows you to override the default behavior is USERELATIONSHIP.

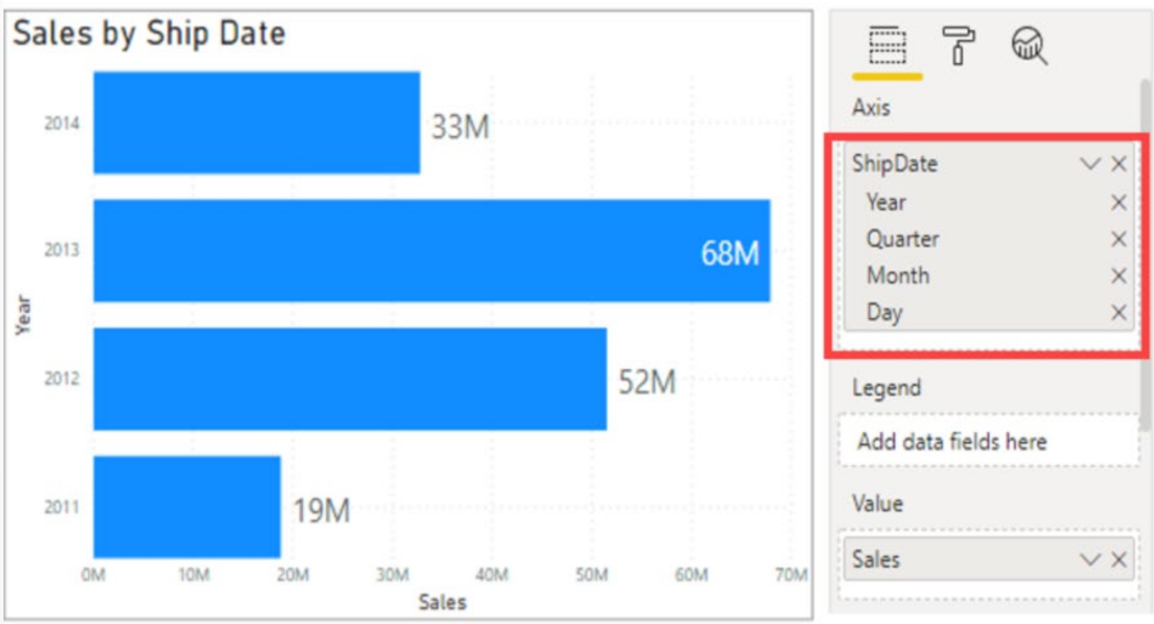
Consider the following data model example.



The preceding screenshot shows an established relationship between the **Date and OrderDate** columns, as shown by the highlighted line connecting the two. The solid line between the two tables indicates that it is the active relationship, meaning that by default, any slicing on the date table where measures in the Sales data are being displayed will be along the **OrderDate** column. A dashed relationship exists between the **Date** and **ShipDate** columns, indicating that it is the inactive relationship. This relationship will never be used unless explicitly declared in a measure.

The goal is to build the following report, where you have two visuals: **Sales by Ship Date** and **Sales by Order Date**.





These visuals show the sales over time, but the top visual is by order date and the bottom is by ship date so, though they are both dates, a different data point is associated with them to get both sets of data on the same visual.

To create this calculated measure for Sales by Ship Date, you can use the DAX function USERELATIONSHIP(). This function is used to specify a relationship to be used in a specific calculation and is done without overriding any existing relationships. It is a beneficial feature in that it allows developers to make additional calculations on inactive relationships by overriding the default active relationship between two tables in a DAX expression, as shown in the following example:

```
Sales by Ship Date = CALCULATE(Sales[TotalPrice], USERELATIONSHIP('Calen-
dar'[Date], Sales[ShipDate]))
```

Now, you will be able to create the bottom visual.

Semi-additive Measures

In situations where you don't want the standard evaluation behavior in Power BI, you can use the CALCULATE and/or USERELATIONSHIP functions. However, more circumstances exist where you don't want the standard behavior. One of those situations is when you have a semi-additive problem to resolve. Standard measures are simple concepts, where they might use the SUM, AVERAGE, MIN, and MAX functions. Thus far, you've been using SUM for the **Total Sales** measure.

Occasionally, summing a measure doesn't make sense, such as when you are performing inventory counts in a warehouse. For example, if on Monday, you have 100 mountain bikes, and on Tuesday you have 125 mountain bikes, you wouldn't want to add those together to indicate that you had 225 mountain bikes between those two days. In this circumstance, if you want to know your stock levels for March, you would need to tell Power BI not to add the measure but instead take the last value for the month of March and assign it to any visual.

You can use the CALCULATE function to complete this action, along with the LastDate function, as shown in the following example:

```
Last Inventory Count =
CALCULATE (
    SUM ( 'Warehouse'[Inventory Count] ),
    LASTDATE ( 'Date'[Date] ))
```

This approach will stop the SUM from crossing all dates. Instead, you will only use the SUM function on the last date of the time period, thus effectively creating a semi-additive measure.

Time-Intelligence

All data analysts will have to deal with time. Dates are important, so we highly recommend that you create or import a dates table. This approach will help make date and time calculations much simpler in DAX.

While some time calculations are simple to do in DAX, others are more difficult. For instance, the following screenshot shows what happens if you want to display a running total.

Month	2014	2015	2016
January		$66,692.8	$100,854.72
February		$107,900	$205,416.67
March		$147,879.9	$315,242.12
April		$203,579.29	$449,872.68
May		$260,402.99	$469,771.34
June		$299,490.99	$469,771.34
July	$30,192.1	$354,955.92	$469,771.34
August	$56,801.5	$404,937.61	$469,771.34
September	$84,437.5	$464,670.63	$469,771.34
October	$125,641.1	$534,999.13	$469,771.34
November	$175,345.1	$580,912.49	$469,771.34
December	$226,298.5	$658,388.75	$469,771.34
Total	**$226,298.5**	**$658,388.75**	**$469,771.34**

Notice that the totals increment for each month but then reset when the year changes. In other programming languages, this result can be fairly complicated, often involving several variables and looping through code. DAX makes this process fairly simple, as shown in the following example:

```
YTD Total Sales = TOTALYTD
(
    SUM('Sales OrderDetails'[Total Price])
    , Dates[Date]
)
```

The **YTD Total Sales** measure uses a built-in DAX function called TOTALYTD. This function takes an argument for the type of calculation. You can use the SUM function to get the Total Price, as you've done throughout this module. The second argument that you want to operate over is the **Dates** field. You can use your Dates table and add this measure to your visual, and you'll get the running total result that you're looking for. You can use all functions with YTD, MTD, and QTD in a similar fashion.

Another example of working with time would be comparing your current sales with the sales of a previous time period. For instance, if you want to see the total sales of the month next to the total sales of the prior month, you would enter the DAX measure definition, as shown in the following example:

```
Total Sales Previous Month = CALCULATE
(
    sum('Sales OrderDetails'[Total Price])
    , PREVIOUSMONTH(Dates[Date])
)
```

This measure uses the CALCULATE function, indicating that you're overriding the context to evaluate this expression the way that you want to. You're summing Total Price, as you've been doing throughout this module. For the second argument, you're using PREVIOUSMONTH for the override, which tells Power BI that, no matter what month is the default, the system should override it to be the previous month.

The following screenshot shows the results in a table visual.

Year	Month	Total Sales	Total Sales Previous Month
2015	March	$39,979.9	$41,207.2
2015	April	$55,699.39	$39,979.9
2015	May	$56,823.7	$55,699.39
2015	June	$39,088	$56,823.7
2015	July	$55,464.93	$39,088
2015	August	$49,981.69	$55,464.93
2015	September	$59,733.02	$49,981.69
2015	October	$70,328.5	$59,733.02
2015	November	$45,913.36	$70,328.5
2015	December	$77,476.26	$45,913.36

When you examine the months side-by-side, notice that the total sales for July compare to the total sales for June.

Module Review

This module started you on a journey to understanding DAX. You learned about creating simple DAX columns and measures, how they work, and how to choose when to do one over the other. You learned about context and how to override it with the CALCULATE function, and you learned about time intelligence and semi-additive measures. Mastery of DAX will take effort and time, but this module has provided you with a great start.

Knowledge Check

Question 1

What type of Measure uses SUM to aggregate over one set of dimensions and a different aggregation over a different set of dimension?

☐ Additive

☐ Aggregate

☐ Semi-additive

Question 2

What type of functions enable you to manipulate data using time periods?

☐ Time intelligence

☐ Comparer functions

☐ Value functions

Question 3

Which two functions will help you compare dates to the previous month?

- ☐ TOTALYTD and PREVIOUSMONTH
- ☐ CALCULATE and TOTALTYD
- ☐ CALCULATE and PREVIOUSMONTH

Answers

Question 1

Which are calculated on demand?

☐ Calculated columns

☐ Calculated tables

■ Measures

Question 2

Which are calculated based on the filters that are used by the report user? Calculated columns or measures?

■ Measures

☐ Calculated columns

Question 1

Which DAX function evaluates an expression in a modified filter context?

☐ SUMX

■ CALCULATE

☐ ALL

Question 2

Why would you want to override the default context?

☐ To create measures that behave according to the user's selection

■ To create measures that behave according to your intentions, regardless of what the user selects

Question 1

What type of Measure uses SUM to aggregate over one set of dimensions and a different aggregation over a different set of dimension?

☐ Additive

☐ Aggregate

■ Semi-additive

Question 2

What type of functions enable you to manipulate data using time periods?

■ Time intelligence

☐ Comparer functions

☐ Value functions

Question 3

Which two functions will help you compare dates to the previous month?

☐ TOTALYTD and PREVIOUSMONTH

☐ CALCULATE and TOTALTYD

■ CALCULATE and PREVIOUSMONTH

Module 6 Optimize Model Performance

Optimize the data model for Performance

Introduction to performance optimization

Performance optimization, also known as performance tuning, involves making changes to the current state of the data model so that it runs more efficiently. Essentially, when your data model is optimized, it performs better.

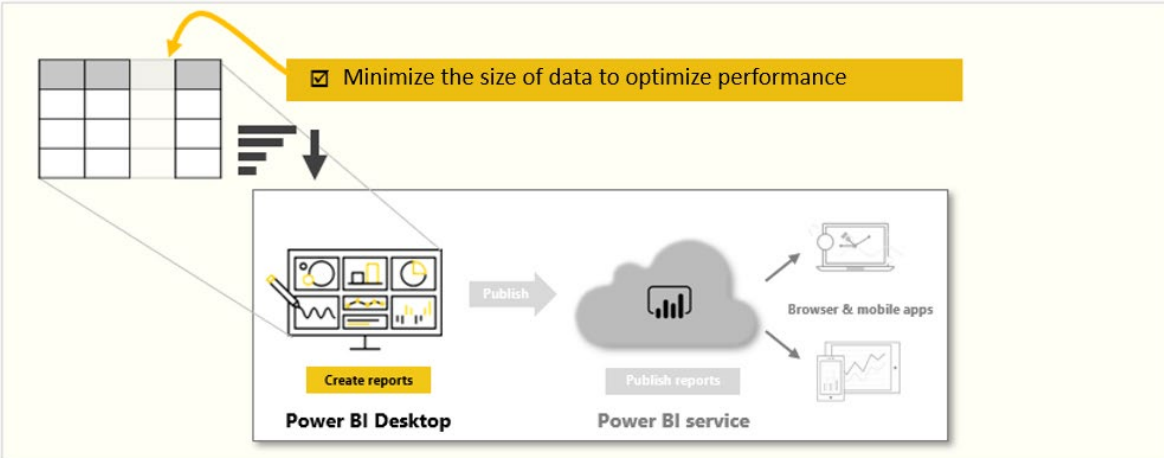
You might find that your report runs well in test and development environments, but when deployed to production for broader consumption, performance issues arise. From a report user's perspective, poor performance is characterized by report pages that take longer to load and visuals taking more time to update. This poor performance results in a negative user experience.

As a data analyst, you will spend approximately 90 percent of your time working with your data, and nine times out of ten, poor performance is a direct result of a bad data model, bad Data Analysis Expressions (DAX), or the mix of the two. The process of designing a data model for performance can be tedious, and it is often underestimated. However, if you address performance issues during development, you will have a robust Power BI data model that will return better reporting performance and a more positive user experience. Ultimately, you will also be able to maintain optimized performance. As your organization grows, the size of its data grows, and its data model becomes more complex. By optimizing your data model early, you can mitigate the negative impact that this growth might have on the performance of your data model.

A smaller sized data model uses less resources (memory) and achieves faster data refresh, calculations, and rendering of visuals in reports. Therefore, the performance optimization process involves minimizing the size of the data model and making the most efficient use of the data in the model, which includes:

- Ensuring that the correct data types are used.
- Deleting unnecessary columns and rows.
- Avoiding repeated values.
- Replacing numeric columns with measures.
- Reducing cardinalities.

- Analyzing model metadata.
- Summarizing data where possible.



In this module, you will be introduced to the steps, processes, and concepts that are necessary to optimize a data model for enterprise-level performance. However, keep in mind that, while the basic performance and best practices guidance in Power BI will lead you a long way, to optimize a data model for query performance, you will likely have to partner with a data engineer to drive data model optimizing in the source data sources.

For example, assume that you work as a Microsoft Power BI developer for Tailwind Traders. You have been given a task to review a data model that was built a few years ago by another developer, a person who has since left the organization.

The data model produces a report that has received negative feedback from users. The users are happy with the results that they see in the report, but they are not satisfied with the report performance. Loading the pages in the report is taking too long, and tables are not refreshing quickly enough when certain selections are made. In addition to this feedback, the IT team has highlighted that the file size of this particular data model is too large, and it is putting a strain on the organization's resources.

You need to review the data model to identify the root cause of the performance issues and make changes to optimization performance.

By the end of this module, you will be able to:

- Review the performance of measures, relationships, and visuals.
- Use variables to improve performance and troubleshooting.
- Improve performance by reducing cardinality levels.
- Optimize DirectQuery models with table level storage.
- Create and manage aggregations.

Use variables to improve performance and troubleshooting

You can use variables in your DAX formulas to help you write less complex and more efficient calculations. Variables are underused by developers who are starting out in Power BI Desktop, but they are effective and you should use them by default when you are creating measures.

Some expressions involve the use of many nested functions and the reuse of expression logic. These expressions take a longer time to process and are difficult to read and, therefore, troubleshoot. If you use variables, you can save query processing time. This change is a step in the right direction toward optimizing the performance of a data model.

The use of variables in your data model provides the following advantages:

- **Improved performance** - Variables can make measures more efficient because they remove the need for Power BI to evaluate the same expression multiple times. You can achieve the same results in a query in about half the original processing time.
- **Improved readability** - Variables have short, self-describing names and are used in place of an ambiguous, multi-worded expression. You might find it easier to read and understand the formulas when variables are used.
- **Simplified debugging** - You can use variables to debug a formula and test expressions, which can be helpful during troubleshooting.
- **Reduced complexity** - Variables do not require the use of EARLIER or EARLIEST DAX functions, which are difficult to understand. These functions were required before variables were introduced, and were written in complex expressions that introduced new filter contexts. Now that you can use variables instead of those functions, you can write fewer complex formulas.

Use variables to improve performance

To illustrate how you can use a variable to make a measure more efficient, the following table displays a measure definition in two different ways. Notice that the formula repeats the expression that calculates "same period last year" but in two different ways: the first instance uses the normal DAX calculation method and the second one uses variables in the calculation.

The second row of the table shows the improved measure definition. This definition uses the VAR keyword to introduce a variable named **SalesPriorYear**, and it uses an expression to assign the "same period last year" result to that new variable. It then uses the variable twice in the RETURN expression.

Without variable

```
Sales YoY Growth =
DIVIDE (
    ( [Sales] - CALCULATE ( [Sales], PARALLELPERIOD ( 'Date'[Date], -12,
MONTH ) ) ),
    CALCULATE ( [Sales], PARALLELPERIOD ( 'Date'[Date], -12, MONTH ) )
)
```

With variable

```
Sales YoY Growth =
VAR SalesPriorYear =
    CALCULATE ( [Sales], PARALLELPERIOD ( 'Date'[Date], -12, MONTH ) )
VAR SalesVariance =
    DIVIDE ( ( [Sales] - SalesPriorYear ), SalesPriorYear )
RETURN
    SalesVariance
```

In the first measure definition in the table, the formula is inefficient because it requires Power BI to evaluate the same expression twice. The second definition is more efficient because, due to the variable, Power BI only needs to evaluate the expression once.

If your data model has multiple queries with multiple measures, the use of variables could cut the overall query processing time in half and improve the overall performance of the data model. Furthermore, this solution is a simple one; imagine the savings as the formulas get more complicated, for instance, when you are dealing with percentages and running totals.

Use variables to improve readability

In addition to improved performance, you might notice how the use of variables makes the code simpler to read.

When using variables, it is best practice to use descriptive names for the variables. In the previous example, the variable is called **SalesPriorYear**, which clearly states what the variable is calculating. Consider the outcome of using a variable that was called **X**, **temp** or **variable1**; the purpose of the variable would not be clear at all.

Using clear, concise, meaningful names will help make it easier for you to understand what you are trying to calculate, and it will be much simpler for other developers to maintain the report in the future.

Use variables to troubleshoot multiple steps

You can use variables to help you debug a formula and identify what the issue is. Variables help simplify the task of troubleshooting your DAX calculation by evaluating each variable separately and by recalling them after the RETURN expression.

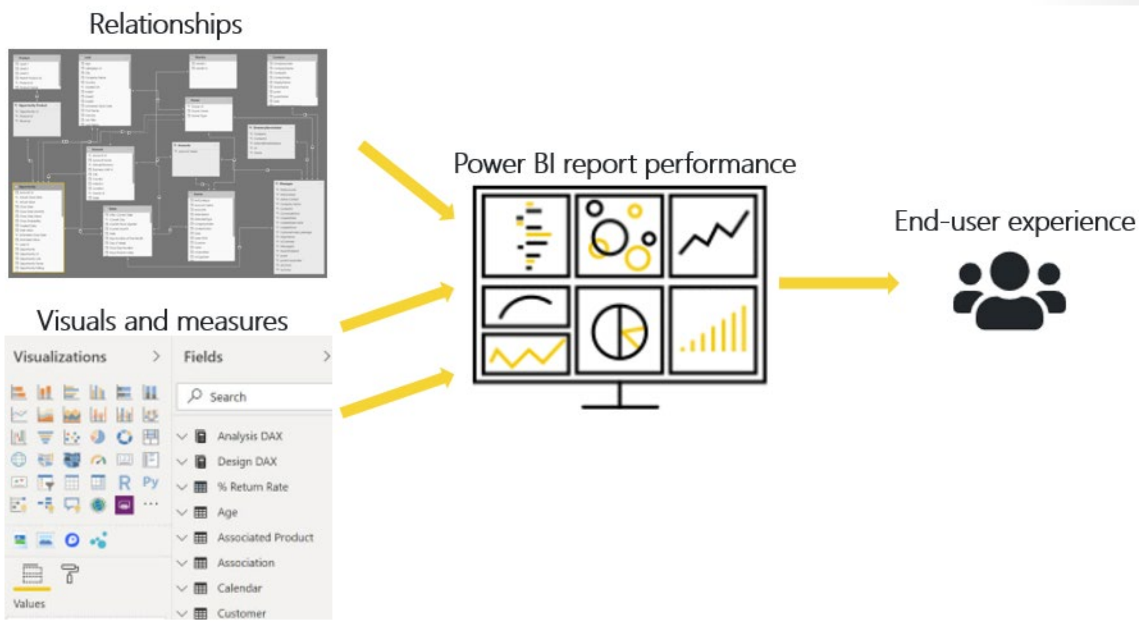
In the following example, you test an expression that is assigned to a variable. You temporarily rewrite the RETURN expression to write to the variable. The measure definition returns only the **SalesPriorYear** variable and it comments-out the intended RETURN expression.

```
Sales YoY Growth % =
VAR SalesPriorYear =  CALCULATE([Sales], PARALLELPERIOD('Date'[Date], -12,
MONTH))
VAR SalesPriorYear% = DIVIDE(([Sales] - SalesPriorYear), SalesPriorYear)
RETURN  SalesPriorYear
```

The RETURN expression will display the **SalesPriorYear** value only. This technique allows you to revert the expression when you have completed the debugging. It also makes calculations simpler to understand due to reduced complexity of the DAX code.

Performance Analyzer

If your data model has multiple tables, complex relationships, intricate calculations, multiple visuals, and redundant data, a potential exists for poor report performance. The poor performance of a report leads to a negative user experience.



To optimize performance, you must first identify where the problem is coming from; in other words, find out which elements of your report and data model are causing the performance issues. Afterward, you can take action to resolve those issues and, therefore, improve performance.

Identify report performance bottlenecks

To achieve optimal performance in your reports, you need to create an efficient data model that has fast running queries and measures. When you have a good foundation, you can improve the model further by analyzing the query plans and dependencies and then making changes to further optimize performance.

You should review the measures and queries in your data model to ensure that you are using the most efficient way to get the results that you want. Your starting point should be to identify bottlenecks that exist in the code. When you identify the slowest query in the data model, you can focus on the biggest bottleneck first and establish a priority list to work through the other issues.

Analyze performance

You can use **Performance analyzer** in Power BI Desktop to help you find out how each of your report elements are performing when users interact with them. For example, you can determine how long it takes for a particular visual to refresh when it is initiated by a user interaction. **Performance analyzer** will help you identify the elements that are contributing to your performance issues, which can be useful during troubleshooting.

Before you run **Performance analyzer**, to ensure you get the most accurate results in your analysis (test), make sure that you start with a clear visual cache and a clear data engine cache.

- **Visual cache** - When you load a visual, you can't clear this visual cache without closing Power BI Desktop and opening it again. To avoid any caching in play, you need to start your analysis with a clean visual cache.

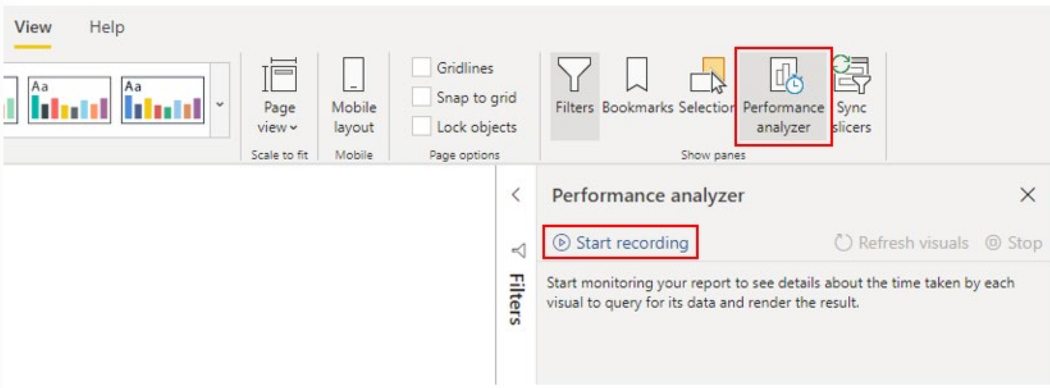
  To ensure that you have a clear visual cache, add a blank page to your Power BI Desktop (.pbix) file and then, with that page selected, save and close the file. Reopen the Power BI Desktop (.pbix) file that you want to analyze. It will open on the blank page.

- **Data engine cache** - When a query is run, the results are cached, so the results of your analysis will be misleading. You need to clear the data cache before rerunning the visual.

  To clear the data cache, you can either restart Power BI Desktop or connect DAX Studio to the data model and then call Clear Cache.

When you have cleared the caches and opened the Power BI Desktop file on the blank page, go to the **View** tab and select the **Performance analyzer** option.

To begin the analysis process, select **Start recording**, select the page of the report that you want to analyze, and interact with the elements of the report that you want to measure. You will see the results of your interactions display in the **Performance analyzer** pane as you work. When you are finished, select the **Stop** button.

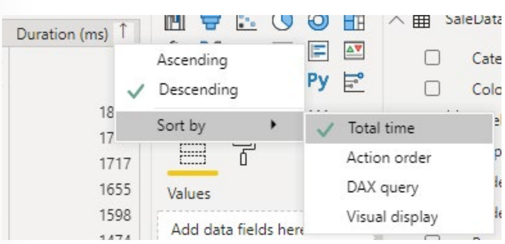


For more detailed information, see **Use Performance Analyzer to examine report element performance**[1].

Review performance results

Review results

You can review the results of your performance test in the **Performance analyzer** pane. To review the tasks in order of duration, longest to shortest, right-click the **Sort** icon next to the **Duration (ms)** column header, and then select **Total time** in **Descending** order.

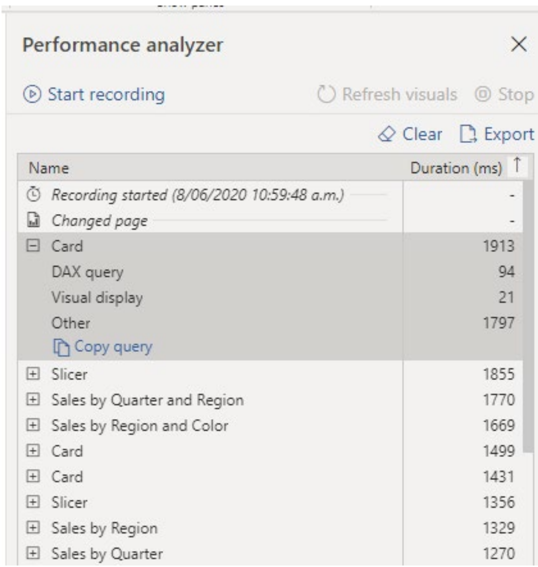


The log information for each visual shows how much time it took (duration) to complete the following categories of tasks:

- **DAX query** - The time it took for the visual to send the query, along with the time it took Analysis Services to return the results.

1 https://docs.microsoft.com/power-bi/create-reports/desktop-performance-analyzer

- **Visual display** - The time it took for the visual to render on the screen, including the time required to retrieve web images or geocoding.
- **Other** - The time it took the visual to prepare queries, wait for other visuals to complete, or perform other background processing tasks. If this category displays a long duration, the only real way to reduce this duration is to optimize DAX queries for other visuals, or reduce the number of visuals in the report.



The results of the analysis test help you to understand the behavior of your data model and identify the elements that you need to optimize. You can compare the duration of each element in the report and identify the elements that have a long duration. You should focus on those elements and investigate why it takes them so long to load on the report page.

To analyze your queries in more detail, you can use DAX Studio, which is a free, open-source tool that is provided by another service.

Data model

If the duration of measures and visuals are displaying low values (in other words they have a short duration time), they are not the reason for the performance issues. Instead, if the DAX query is displaying a high duration value, it is likely that a measure is written poorly or an issue has occurred with the data model. The issue might be caused by the relationships, columns, or metadata in your model, or it could be the status of the **Auto date/time** option, as explained in the following section.

Relationships

You should review the relationships between your tables to ensure that you have established the correct relationships. Check that relationship cardinality properties are correctly configured. For example, a one-side column that contains unique values might be incorrectly configured as a many-side column. You will learn more about how cardinality affects performance later in this module.

Columns

It is best practice to not import columns of data that you do not need. To avoid deleting columns in Power Query Editor, you should try to deal with them at the source when loading data into Power BI

Desktop. However, if it is impossible to remove redundant columns from the source query or the data has already been imported in its raw state, you can always use Power Query Editor to examine each column. Ask yourself if you really need each column and try to identify the benefit that each one adds to your data model. If you find that a column adds no value, you should remove it from your data model. For example, suppose that you have an ID column with thousands of unique rows. You know that you won't use this particular column in a relationship, so it will not be used in a report. Therefore, you should consider this column as unnecessary and admit that it is wasting space in your data model.

When you remove an unnecessary column, you will reduce the size of the data model which, in turn, results in a smaller file size and faster refresh time. Also, because the dataset contains only relevant data, the overall report performance will be improved.

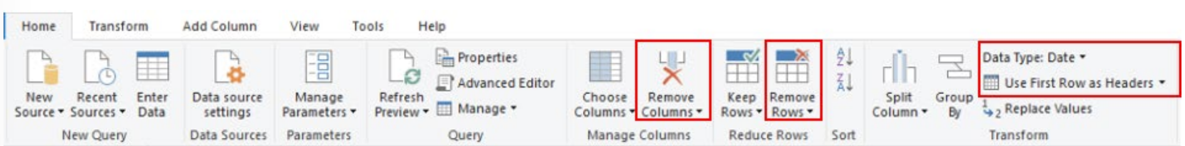
For more information, see **Data reduction techniques for Import modeling**[2].

Metadata

Metadata is information about other data. Power BI metadata contains information on your data model, such as the name, data type and format of each of the columns, the schema of the database, the report design, when the file was last modified, the data refresh rates, and much more.

When you load data into Power BI Desktop, it is good practice to analyze the corresponding metadata so you can identify any inconsistences with your dataset and normalize the data before you start to build reports. Running analysis on your metadata will improve data model performance because, while analyzing your metadata, you will identify unnecessary columns, errors within your data, incorrect data types, the volume of data being loaded (large datasets, including transactional or historic data, will take longer to load), and much more.

You can use Power Query Editor in Power BI Desktop to examine the columns, rows, and values of the raw data. You can then use the available tools, such as those highlighted in the following screenshot, to make the necessary changes.

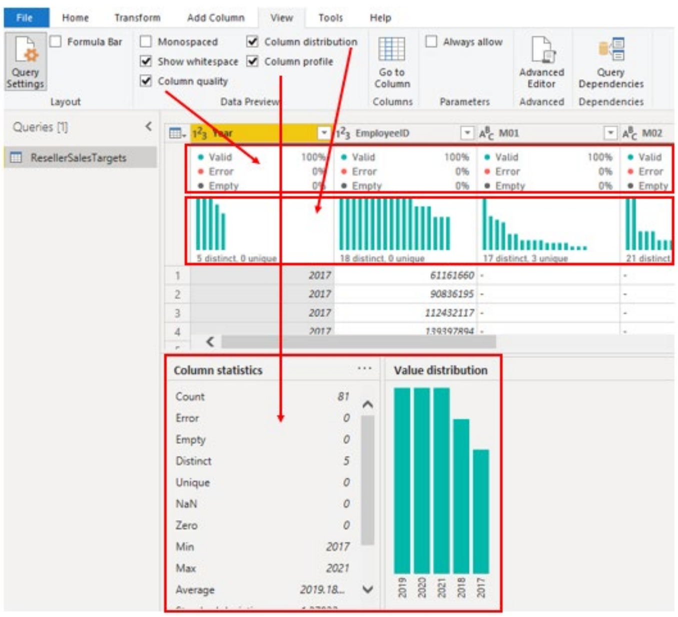


The Power Query options include:

- **Unnecessary columns** - Evaluates the need for each column. If one or more columns will not be used in the report and are therefore unnecessary, you should remove them by using the **Remove Columns** option on the **Home** tab.
- **Unnecessary rows** - Checks the first few rows in the dataset to see if they are empty or if they contain data that you do not need in your reports; if so, it removes those rows by using the **Remove Top Rows** option on the **Home** tab.
- **Data type** - Evaluates the column data types to ensure that each one is incorrect. If you identify a data type that is incorrect, change it by selecting the column, selecting **Data Type** on the **Transform** tab, and then selecting the correct data type from the list.
- **Query names** - Examines the query (table) names in the **Queries** pane. Just like you did for column header names, you should change uncommon or unhelpful query names to names that are more obvious or names that the user is more familiar with. You can rename a query by right-clicking that query, selecting **Rename**, editing the name as required, and then pressing **Enter**.

2 https://docs.microsoft.com/power-bi/guidance/import-modeling-data-reduction

- **Column details** - Power Query Editor has the following three data preview options that you can use to analyze the metadata that is associated with your columns. You can find these options on the **View** tab, as illustrated in the following screenshot.
  - **Column quality** - Determines what percentage of items in the column are valid, have errors, or are empty. If the Valid percentage is not 100, you should investigate the reason, correct the errors, and populate empty values.
  - **Column distribution** - Identifies how many distinct items you have and how many are unique. This information is useful when you want to identify the cardinality of a column. You will investigate this further later in this module.
  - **Column profile** - Shows more statistics for the column and a chart showing the distribution of the unique items.



NOTE: If you are reviewing a large dataset with more than 1,000 rows, and you want to analyze that whole dataset, you need to change the default option at the bottom of the window. Select **Column profiling based on top 1000 rows>Column profiling based on entire data set**.

Other metadata that you should consider is the information about the data model as a whole, such as the file size and data refresh rates. You can find this metadata in the associated Power BI Desktop (.pbix) file. The data that you load into Power BI Desktop is compressed and stored to the disk by the VertiPaq storage engine. The size of your data model has a direct impact on its performance; a smaller sized data model uses less resources (memory) and achieves faster data refresh, calculations, and rendering of visuals in reports.

Auto date/time feature

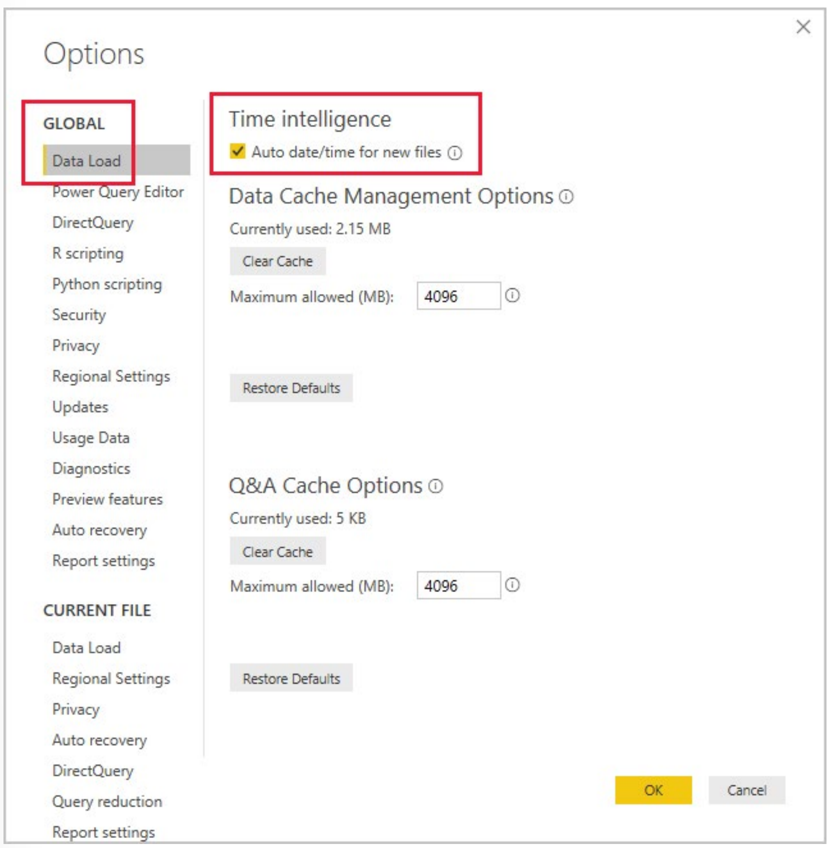
Another item to consider when optimizing performance is the **Auto date/time** option in Power BI Desktop. By default, this feature is enabled globally, which means that Power BI Desktop automatically creates a hidden calculated table for each date column, provided that certain conditions are met. The new, hidden tables are in addition to the tables that you already have in your dataset.

The **Auto date/time** option allows you to work with time intelligence when filtering, grouping, and drilling down through calendar time periods. We recommend that you keep the **Auto date/time** option enabled only when you work with calendar time periods and when you have simplistic model requirements in relation to time.

If your data source already defines a date dimension table, that table should be used to consistently define time within your organization, and you should disable the global **Auto date/time** option. Disabling this option can lower the size of your data model and reduce the refresh time.

You can enable/disable this **Auto date/time** option globally so that it applies to all of your Power BI Desktop files, or you can enable/disable the option for the current file so that it applies to an individual file only.

To enable/disable this **Auto date/time** option, go to **File>Options and settings>Options**, and then select either the **Global** or **Current File** page. On either page, select **Data Load** and then, in the **Time Intelligence** section, select or clear the check box as required.



For an overview and general introduction to the **Auto date/time** feature, see **Apply auto date/time in Power BI Desktop**[3].

3 https://docs.microsoft.com/power-bi/transform-model/desktop-auto-date-time

Analyze query plans

DAX query

When you examine the results in the **Performance analyzer** pane, you can see how long it took the Power BI Desktop engine to evaluate each query (in milliseconds). A good starting point is any DAX query that is taking longer than 120 milliseconds. In this example, you identify one particular query that has a large duration time.

Performance analyzer highlights potential issues but does not tell you what needs to be done to improve them. You might want to conduct further investigation into why this measure takes so long to process. You can use DAX Studio to investigate your queries in more detail.

For example, select **Copy Query** to copy the calculation formula onto the clipboard, then paste it into Dax Studio. You can then review the calculation step in more detail. In this example, you are trying to count the total number of products with order quantities greater than or equal to five.

```
Count Customers =
CALCULATE (
    DISTINCTCOUNT ( Order[ProductID] ),
    FILTER ( Order, Order[OrderQty] >= 5 )
)
```

After analyzing the query, you can use your own knowledge and experience to identify where the performance issues are. You can also try using different DAX functions to see if they improve performance. In the following example, the FILTER function was replaced with the KEEPFILTER function. When the test was run again in **Performance analyzer**, the duration was shorter as a result of the KEEPFILTER function.

```
Count Customers =
CALCULATE (
    DISTINCTCOUNT ( Order[ProductID] ),
    KEEPFILTERS (Order[OrderQty] >= 5 )
)
```

In this case, you can replace the FILTER function with the KEEPFILTER function to significantly reduce the evaluation duration time for this query. When you make this change, to check whether the duration time has improved or not, clear the data cache and then rerun the **Performance analyzer** process.

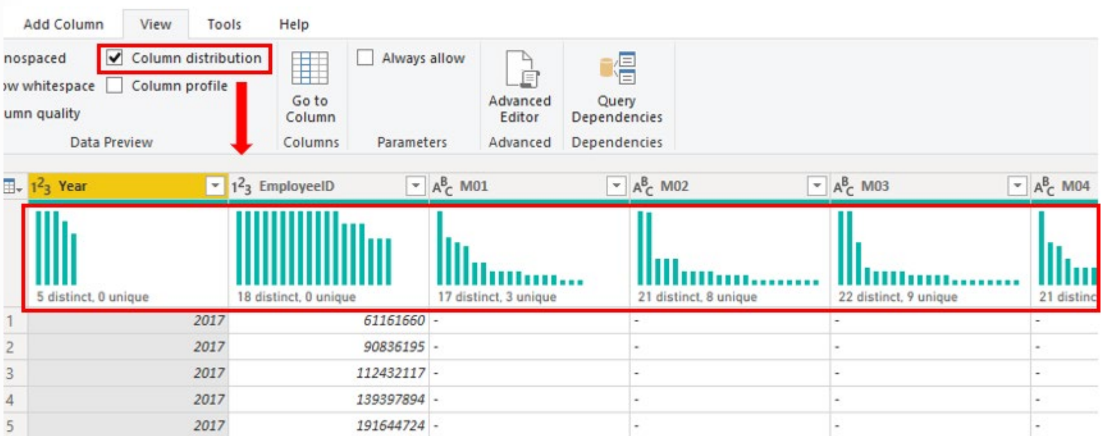
Reduce Cardinality

Cardinality is a term that is used to describe the uniqueness of the values in a column. Cardinality is also used in the context of the relationships between two tables, where it describes the direction of the relationship.

Identify cardinality levels in columns

Previously, when you used Power Query Editor to analyze the metadata, the **Column distribution** option on the **View** tab displayed statistics on how many distinct and unique items were in each column in the data.

- **Distinct values count** - The total number of different values found in a given column.
- **Unique values count** - The total number of values that only appear once in a given column.



A column that has a lot of repeated values in its range (distinct count is high) will have a low level of cardinality. Conversely, a column that has a lot of unique values in its range (unique count is high) will have a high level of cardinality.

Lower cardinality leads to more optimized performance, so you might need to reduce the number of high cardinally columns in your dataset.

Reduce relationship cardinality

When you import multiple tables, it is possible that you'll do some analysis by using data from all those tables. Relationships between those tables are necessary to accurately calculate results and display the correct information in your reports. Power BI Desktop helps make creating those relationships easier. In fact, in most cases, you won't have to do anything, the autodetect feature does it for you. However, you might occasionally have to create relationships or need to make changes to a relationship. Regardless, it's important to understand relationships in Power BI Desktop and how to create and edit them.

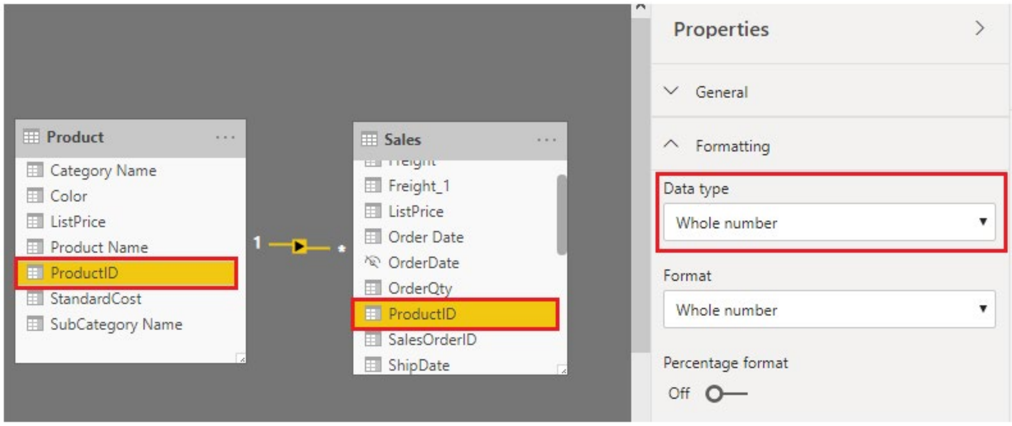
When you create or edit a relationship, you can configure additional options. By default, Power BI Desktop automatically configures additional options based on its best guess, which can be different for each relationship based on the data in the columns.

The relationships can have different cardinality. Cardinality is the direction of the relationship, and each model relationship must be defined with a cardinality type. The cardinality options in Power BI are:

- **Many-to-one (*:1)** - This relationship is the most common, default type. It means that 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.
- **One-to-one (1:1)** - In this relationship type, the column in one table has only one instance of a particular value, and the other related table has only one instance of a particular value.
- **One-to-many (1:*)** - In this relationship type, the column in one table has only one instance of a particular value, and the other related table can have more than one instance of a value.
- **Many-to-many (* : *)** - With composite models, you can establish a many-to-many relationship between tables, which removes requirements for unique values in tables. It also removes previous workarounds, such as introducing new tables only to establish relationships.

During development, you will be creating and editing relationships in your model, so when you are building new relationships in your model, regardless of what cardinality you have chosen, always ensure that both of the columns that you are using to participate in a relationship are sharing the same data type. Your model will never work if you try to build a relationship between two columns, where one column has a text data type and another column has an integer data type.

In the following example, the **ProductID** field has the data type **Whole number** in the Product and Sales tables. The columns with data type **Integer** perform better than columns with data type **Text**.



Improve performance by reducing cardinality levels

Power BI Desktop offers different techniques that you can use to help reduce the data that is loaded into data models, such as summarization. Reducing the data that is loaded into your model will improve the relationship cardinality of the report. For this reason, it is important that you strive to minimize the data that will be loaded into your models. This case is especially true for large models, or models that you anticipate will grow to become large over time.

Perhaps the most effective technique to reduce a model size is to use a summary table from the data source. Where a detail table might contain every transaction, a summary table would contain one record per day, per week, or per month. It might be an average of all of the transactions per day, for instance.

For example, a source sales fact table stores one row for each order line. Significant data reduction could be achieved by summarizing all sales metrics if you group by date, customer, and product, and individual transaction detail is not needed.

Consider, then, that an even more significant data reduction could be achieved by grouping by date at month level. It could achieve a possible 99 percent reduction in model size; but, reporting at day level or an individual order level is no longer possible. Deciding to summarize fact-type data will always involve a tradeoff with the detail of your data. A disadvantage is that you may lose the ability to drill into data because the detail no longer exists. This tradeoff could be mitigated by using a mixed model design.

In Power BI Desktop, a Mixed mode design produces a composite model. Essentially, it allows you to determine a storage mode for each table. Therefore, each table can have its **Storage Mode** property set as **Import** or **DirectQuery**.

An effective technique to reduce the model size is to set the **Storage Mode** property for larger fact-type tables to **DirectQuery**. This design approach can work well in conjunction with techniques that are used to summarize your data. For example, the summarized sales data could be used to achieve high performance "summary" reporting. A drill-through page could be created to display granular sales for specific (and narrow) filter context, displaying all in-context sales orders. The drill-through page would include visuals based on a DirectQuery table to retrieve the sales order data (sales order details).

For more information, see **Data reduction techniques for Import modeling**[4].

Implement table granularity

Data granularity is the detail that is represented within your data, meaning that the more granularity your data has, the greater the level of detail within your data.

Data granularity is an important topic for all data analysts, regardless of the Power BI tools that you are using. Defining the correct data granularity can have a big impact on the performance and usability of your Power BI reports and visuals.

Data granularity defined

Consider a scenario where your company manages 1,000 refrigerated semi-trucks. Every few minutes, each truck uses a Microsoft Azure IoT application to record its current temperature. This temperature is important to your organization because, if the refrigeration were to malfunction, it could spoil the entire load, costing thousands of dollars. With so many trucks and so many sensors, extensive data is generated every day. Your report users don't want to sift through numerous records to find the ones that they are particularly interested in.

How can you change the granularity of the data to make the dataset more usable?

In this scenario, you might want to import the data by using a daily average for each truck. That approach would reduce the records in the database to one record for each truck for each day. If you decide that the approach was acceptable enough for tracking costs and errors, then you could use that data granularity. Alternatively, you could select the last recorded temperature, or you could only import records that are above or below a normal range of temperatures. Any of these methods will reduce the total records that you import, while still bringing in data that is comprehensive and valuable.

For different scenarios, you could settle on data granularity that is defined weekly, monthly, or quarterly. Generally, the fewer the records that you are working with, the faster your reports and visuals will function. This approach translates to a faster refresh rate for the entire dataset, which might mean that you can refresh more frequently.

However, that approach has a drawback. If your users want to drill into every single transaction, summarizing the granularity will prevent them from doing that, which can have a negative impact on the user

4 https://docs.microsoft.com/power-bi/guidance/import-modeling-data-reduction#group-by-and-summarize

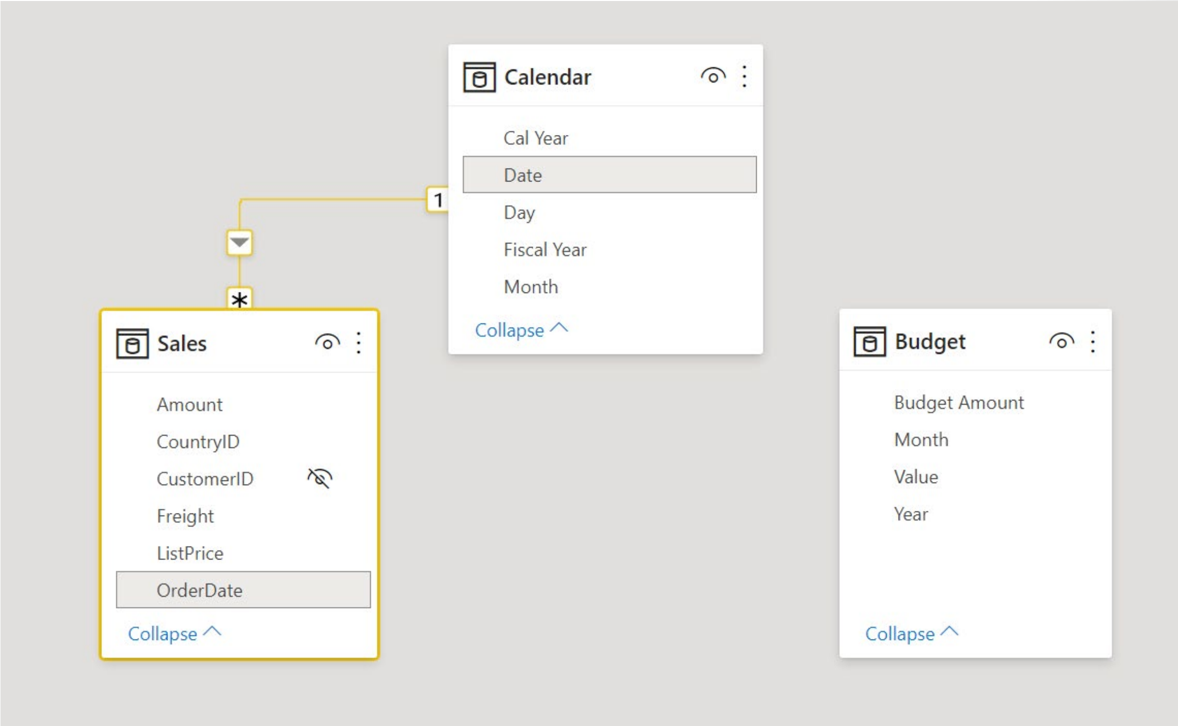
experience. It is important to negotiate the level of data granularity with report users so they understand the implications of these choices.

Change data granularity to build a relationship between two tables

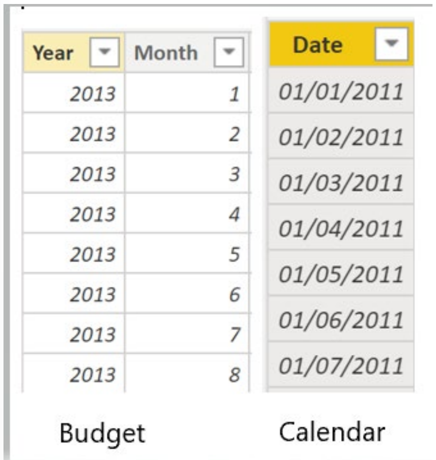
Data granularity can also have an impact when you are building relationships between tables in Power BI.

For example, consider that you are building reports for the Sales team at Tailwind Traders. You have been asked to build a matrix of total sales and budget over time by using the Calendar, Sales, and Budget tables. You notice that the lowest level of time-based detail that the Sales table goes into is by day, for instance 5/1/2020, 6/7/2020, and 6/18/2020. The Budget table only goes to the monthly level, for instance, the budget data is 5/2020 and 6/2020.These tables have different granularities that need to be reconciled before you can build a relationship between tables.

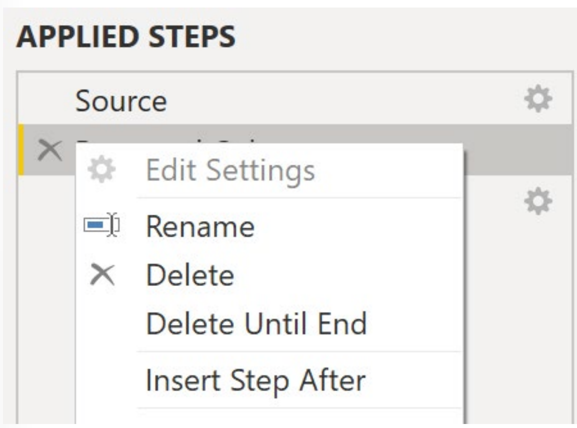
The following figure shows your current data model.



As shown in the preceding figure, a relationship between Budget and Calendar is missing. Therefore, you need to create this relationship before you can build your visual. Notice that if you transform the **Year** and **Month** columns in the Budget table, you can match the format of the **Date** column in the Calendar table. Then, you can establish a relationship between the two columns. To complete this task, you will concatenate the **Year** and **Month** columns and then change the format.



Select **Transform Data** on the ribbon. On **Applied Steps**, on the right pane, right-click the last step and then select **Insert Step After**.



Under **Add Column** on the Home ribbon, select **Custom Column**. Enter the following equation, which will concatenate the **Year** and **Month** columns, and then add a dash in between the column names.

```
Column = Table.AddColumn(#"Renamed Columns", "Custom", each [Year] & "-"
&[Month])
```

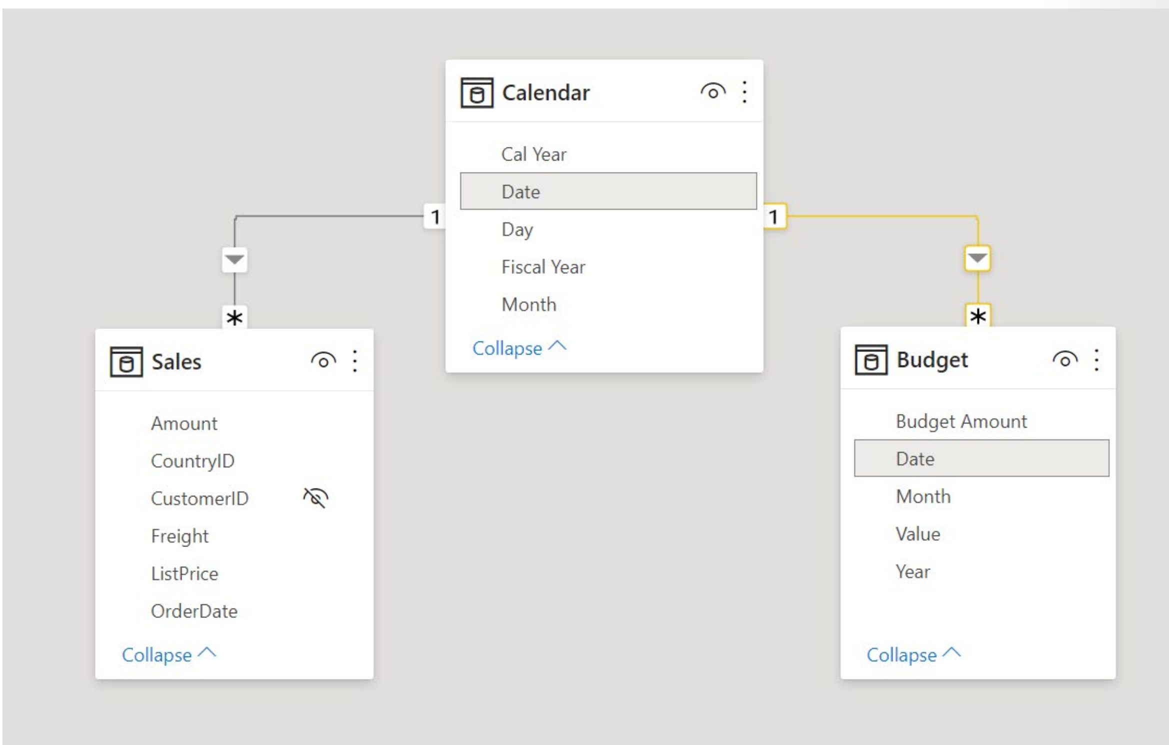
Change the data type to **Date** and then rename the column. Your Budget table should resemble the following figure.

Year	Month	MarketingAmount	Date
2013	1	5000	01/01/2013
2013	2	4000	01/02/2013
2013	3	6000	01/03/2013
2013	4	8000	01/04/2013
2013	5	10000	01/05/2013
2013	6	3500	01/06/2013
2013	7	6500	01/07/2013
2013	8	8500	01/08/2013
2013	9	9000	01/09/2013
2013	10	4000	01/10/2013
2013	11	2000	01/11/2013
2013	12	9500	01/12/2013
2014	1	6500	01/01/2014
2014	2	6000	01/02/2014
2014	3	5000	01/03/2014

Now, you can create a relationship between the Budget and the Calendar tables.

Create a relationship between Calendar and Budget tables

Power BI automatically detects relationships, but you can also go to **Manage Relationships > New** and create the relationship on the **Date** column. The relationship should resemble the following figure.

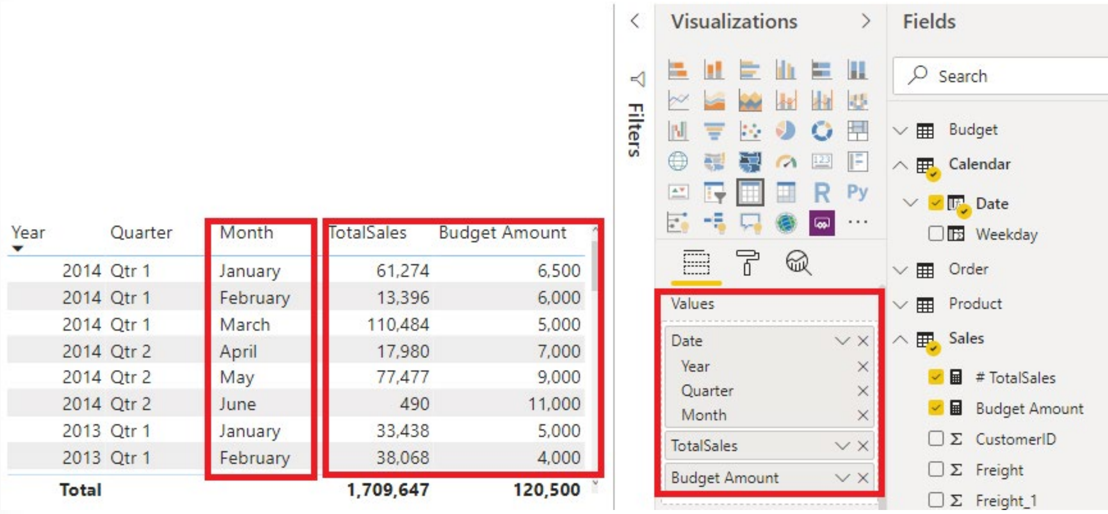


By completing this task, you have ensured that the granularity is the same between your different tables. Now, you need to create DAX measures to calculate **Total Sales** and **BudgetAmount**. Go to the **Data** pane on Power BI Desktop, select **New Measure**, and then create two measures with the following equations:

```
TotalSales = SUM(Sales[Total Sales])/100
```

```
BudgetAmount = SUM (Budget[BudgetAmount])
```

Select the matrix visual on the **Visualization** pane, and then enter these measures and the **Date** into the **Values** field. You have now accomplished the goal of building a matrix of the total sales and budgets over time.



Knowledge Check

Question 1

What benefit do you get from analyzing metadata?

☐ The benefit of analyzing metadata is that you can clearly identify data inconsistencies with your dataset.

☐ The benefit of analyzing the metadata is to get familiar with your data.

☐ The benefit of analyzing the metadata is to know the number of rows, columns and tables being loaded into your model.

Question 2

Which tool enables you to identify bottlenecks that exist in code?

☐ Q&A.

☐ Column profiling.

☐ Performance analyzer.

Question 3

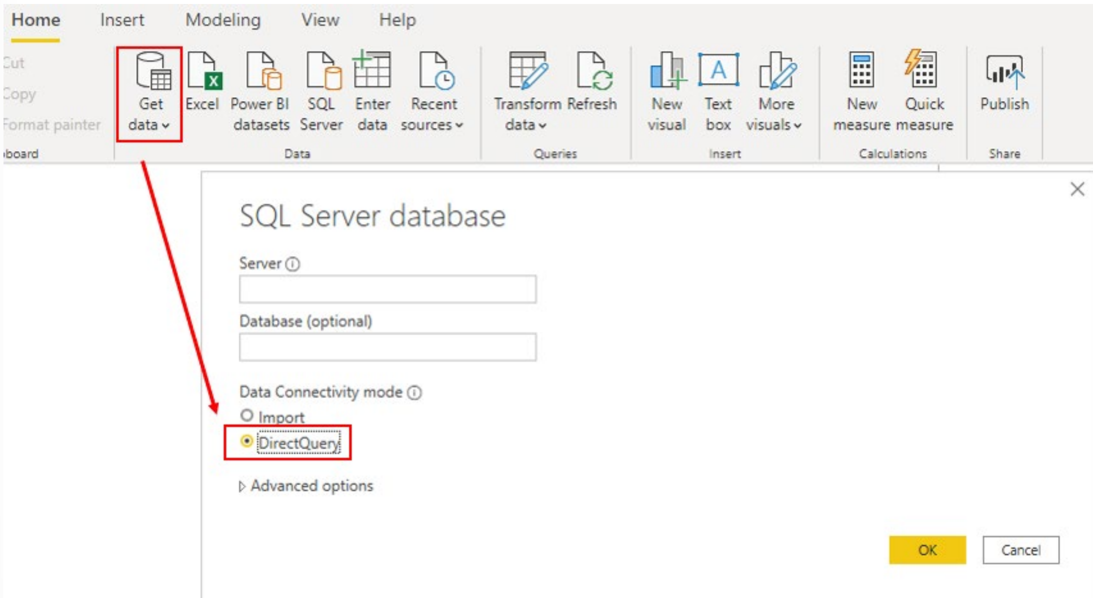
What is cardinality?

- ☐ Cardinality is the granularity of the data.
- ☐ Cardinality is how long it takes for the data to load.
- ☐ Cardinality is a type of visual element.
- ☐ Cardinality is a term that is used to describe the uniqueness of the values in a column. Relationship cardinality refers to the number of rows from one table that are related to another (one to one, one to many, many to many).

Optimize DirectQuery Models

Introduction to DirectQuery

DirectQuery is one way to get data into Power BI Desktop. The DirectQuery method involves connecting directly to data in its source repository from within Power BI Desktop. It is an alternative to importing data into Power BI Desktop.



When you use the DirectQuery method, the overall user experience depends heavily on the performance of the underlying data source. Slow query response times will lead to a negative user experience and, in the worst-case scenarios, queries might time out. Also, the number of users who are opening the reports at any one time will impact the load that is placed on the data source. For example, if your report has 20 visuals in it and 10 people are using the report, 200 queries or more will exist on the data source because each visual will issue one or more queries.

Unfortunately, the performance of your Power BI model will not only be impacted by the performance of the underlying data source, but also by other uncontrollable factors, such as:

- Network latency; faster networks return data quicker.
- The performance of the data source's server and how many other workloads are on that server. For example, consider the implications of a server refresh taking place while hundreds of people are using the same server for different reasons.

Therefore, using DirectQuery poses a risk to the quality of your model's performance. To optimize performance in this situation, you need to have control over, or access to, the source database.

For more detailed information, see **DirectQuery model guidance in Power BI Desktop**[5].

5 https://docs.microsoft.com/power-bi/guidance/directquery-model-guidance

Implications of using DirectQuery

It is best practice to import data into Power BI Desktop, but your organization might need to use the DirectQuery data connectivity mode because of one of the following reasons (benefits of DirectQuery):

- It is suitable in cases where data changes frequently and near real-time reporting is required.
- It can handle large data without the need to pre-aggregate.
- It applies data sovereignty restrictions to comply with legal requirements.
- It can be used with a multidimensional data source that contains measures such as SAP Business Warehouse (BW).

If your organization needs to use DirectQuery, you should clearly understand its behavior within Power BI Desktop and be aware of its limitations. You will then be in a good position to take action to optimize the DirectQuery model as much as possible.

Behavior of DirectQuery connections

When you use DirectQuery to connect to data in Power BI Desktop, that connection behaves in the following way:

- When you initially use the **Get Data** feature in Power BI Desktop, you will select the source. If you connect to a relational source, you can select a set of tables and each one will define a query that logically returns a set of data. If you select a multidimensional source, such as SAP BW, you can only select the source.
- When you load the data, no data is imported into the Power BI Desktop, only the schema is loaded. When you build a visual within Power BI Desktop, queries are sent to the underlying source to retrieve the necessary data. The time it takes to refresh the visual depends on the performance of the underlying data source.
- If changes are made to the underlying data, they won't be immediately reflected in the existing visuals in Power BI due to caching. You need to carry out a refresh to see those changes. The necessary queries are resent for each visual, and the visuals are updated accordingly.
- When you publish the report to the Power BI service, it will result in a dataset in Power BI service, the same as for import. However, no data is included with that dataset.
- When you open an existing report in Power BI service, or build a new one, the underlying source is again queried to retrieve the necessary data. Depending on the location of the original source, you might have to configure an on-premises data gateway.
- You can pin visuals, or entire report pages, as dashboard tiles. The tiles are automatically refreshed on a schedule, for example, every hour. You can control the frequency of this refresh to meet your requirements. When you open a dashboard, the tiles reflect the data at the time of the last refresh and might not include the latest changes that are made to the underlying data source. You can always refresh an open dashboard to ensure that it's up-to-date.

Limitations of DirectQuery connections

The use of DirectQuery can have negative implications. The limitations vary, depending on the specific data source that is being used. You should take the following points into consideration:

- **Performance** - As previously discussed, your overall user experience depends heavily on the performance of the underlying data source.

- **Security** - If you use multiple data sources in a DirectQuery model, it is important to understand how data moves between the underlying data sources and the associated security implications. You should also identify if security rules are applicable to the data in your underlying source because, in Power BI, every user can see that data.
- **Data transformation** - Compared to imported data, data that is sourced from DirectQuery has limitations when it comes to applying data transformation techniques within Power Query Editor. For example, if you connect to an OLAP source, such as SAP BW, you can't make any transformations at all; the entire external model is taken from the data source. If you want to make any transformations to the data, you will need to do this in the underlying data source.
- **Modeling** - Some of the modeling capabilities that you have with imported data aren't available, or are limited, when you use DirectQuery.
- **Reporting** - Almost all the reporting capabilities that you have with imported data are also supported for DirectQuery models, provided that the underlying source offers a suitable level of performance. However, when the report is published in Power BI service, the Quick Insights and Q&A features are not supported. Also, the use of the Explore feature in Excel will likely result in poorer performance.

For more detailed information on the limitations of using DirectQuery, see Implications of using DirectQuery.

Now that you have a brief understanding of how DirectQuery works and the limitations that it poses, you can take action to improve the performance.

Optimize performance

Continuing with the Tailwind Traders scenario, during your review of the data model, you discover that the query used DirectQuery to connect Power BI Desktop to the source data. This use of DirectQuery is the reason why users are experiencing poor report performance. It's taking too long to load the pages in the report, and tables are not refreshing quickly enough when certain selections are made. You need to take action to optimize the performance of the DirectQuery model.

You can examine the queries that are being sent to the underlying source and try to identify the reason for the poor query performance. You can then make changes in Power BI Desktop and the underlying data source to optimize overall performance.

Optimize data in Power BI Desktop

When you have optimized the data source as much as possible, you can take further action within Power BI Desktop by using **Performance analyzer**, where you can isolate queries to validate query plans.

You can analyze the duration of the queries that are being sent to the underlying source to identify the queries that are taking a long time to load. In other words, you can identify where the bottlenecks exist.

You don't need to use a special approach when optimizing a DirectQuery model; you can apply the same optimization techniques that you used on the imported data to tune the data from the DirectQuery source. For example, you can reduce the number of visuals on the report page or reduce the number of fields that are used in a visual. You can also remove unnecessary columns and rows.

For more detailed guidance on how a optimize a DirectQuery query, see: **DirectQuery model guidance in Power BI Desktop**[6] and **Guidance for using DirectQuery successfully**[7].

6 https://docs.microsoft.com/power-bi/guidance/directquery-model-guidance
7 https://docs.microsoft.com/power-bi/connect-data/desktop-directquery-about#guidance-for-using-directquery-successfully

Optimize the underlying data source (connected database)

Your first stop is the data source. You need to tune the source database as much as possible because anything you do to improve the performance of that source database will in turn improve Power BI DirectQuery. The actions that you take in the database will do the most good.

Consider the use of the following standard database practices that apply to most situations:

- Avoid the use of complex calculated columns because the calculation expression will be embedded into the source queries. It is more efficient to push the expression back to the source because it avoids the push down. You could also consider adding surrogate key columns to dimension-type tables.
- Review the indexes and verify that the current indexing is correct. If you need to create new indexes, ensure that they are appropriate.

Refer to the guidance documents of your data source and implement their performance recommendations.

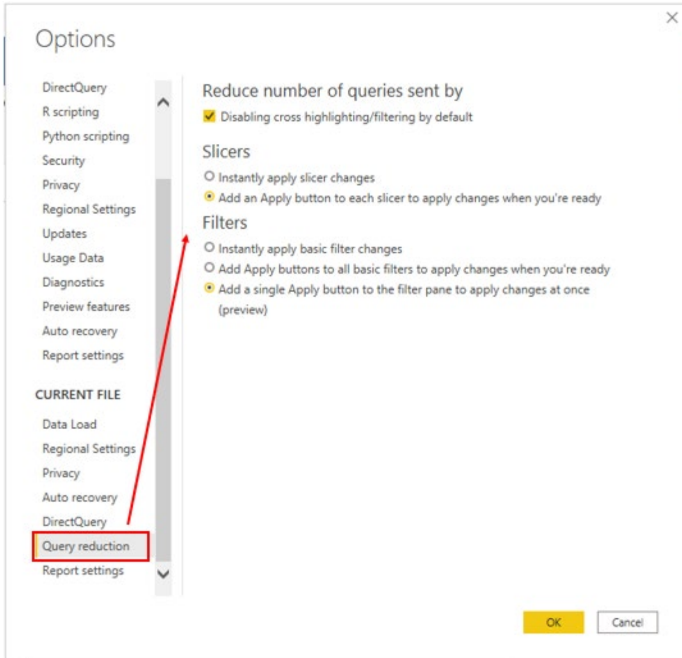
Customize the Query reduction options

Power BI Desktop gives you the option to send fewer queries and to disable certain interactions that will result in a poor experience if the resulting queries take a long time to run. Applying these options prevents queries from continuously hitting the data source, which should improve performance.

In this example, you edit the default settings to apply the available data reduction options to your model. You access the settings by selecting **File>Options and settings>Options**, scrolling down the page, and then selecting the **Query reduction** option.

The following query reduction options are available:

- **Reduce number of queries sent by** - By default, every visual interacts with every other visual. Selecting this check box disables that default interaction. You can then optionally choose which visuals interact with each other by using the **Edit interactions** feature.
- **Slicers** - By default, the **Instantly apply slicer changes** option is selected. To force the report users to manually apply slicer changes, select the **Add an apply button to each slicer to apply changes when you're ready** option.
- **Filters** - By default, the **Instantly apply basic filter changes** option is selected. To force the report users to manually apply filter changes, select one of the alternative options:
    - **Add an apply button to all basic filters to apply changes when you're ready**
    - **Add a single apply button to the filter pane to apply changes at once (preview)**



Knowledge Check

Question 1

Which Power BI option gives you the option to send fewer queries and disable certain interactions?

☐ Direct query

☐ Query reduction

☐ Query diagnostics

Question 2

Other than Power BI, another place for performance optimization can be performed is where?

☐ At the data source

☐ In the Power BI service

☐ In Microsoft SharePoint

Question 3

Is it possible to create a relationship between two columns if they are different DATA TYPE columns?

☐ Yes, if cardinality of the relationship is set to Many-to-Many.

☐ Yes, the above is fully supported in latest version of Power BI desktop.

☐ No, both columns in a relationship must be sharing the same DATA TYPE.

Answers

Question 1

What benefit do you get from analyzing metadata?

■ The benefit of analyzing metadata is that you can clearly identify data inconsistencies with your dataset.

□ The benefit of analyzing the metadata is to get familiar with your data.

□ The benefit of analyzing the metadata is to know the number of rows, columns and tables being loaded into your model.

Question 2

Which tool enables you to identify bottlenecks that exist in code?

□ Q&A.

□ Column profiling.

■ Performance analyzer.

Question 3

What is cardinality?

□ Cardinality is the granularity of the data.

□ Cardinality is how long it takes for the data to load.

□ Cardinality is a type of visual element.

■ Cardinality is a term that is used to describe the uniqueness of the values in a column. Relationship cardinality refers to the number of rows from one table that are related to another (one to one, one to many, many to many).

Question 1

Which Power BI option gives you the option to send fewer queries and disable certain interactions?

□ Direct query

■ Query reduction

□ Query diagnostics

Question 2

Other than Power BI, another place for performance optimization can be performed is where?

■ At the data source

□ In the Power BI service

□ In Microsoft SharePoint

Question 3

Is it possible to create a relationship between two columns if they are different DATA TYPE columns?

☐ Yes, if cardinality of the relationship is set to Many-to-Many.

☐ Yes, the above is fully supported in latest version of Power BI desktop.

■ No, both columns in a relationship must be sharing the same DATA TYPE.

Module 7 Create Reports

Design a Report

Introduction

Power BI visuals are attractive charts and graphics that you can use to revitalize your data. Visuals allow you to share data insights more effectively and increase comprehension, retention, and appeal. Visuals are a fundamental part of your report because they help your report audience connect and interact with the information to make informed decisions quickly.

After you've loaded and modeled your organization's data in Power BI Desktop, you will be ready to start creating your reports. In this module, you'll use the report editor in Power BI Desktop to add suitable visuals to your report canvas. You'll then customize those visuals to meet your organization's requirements.

Consider a scenario where you work as a Power BI developer for Tailwind Traders. Your organization wants to transform the way that it presents its data to management and stakeholders. It wants to replace the current text and tabular report format with a more visual approach so that users will find the reports more interesting. Additionally, by using a visual approach, the company can provide users with quicker, easier access to the information that they need to make their business decisions. You are tasked with creating a Power BI report that is based on a combination of visuals that are customized to match your organization's branding and report presentation requirements.

By the end of this module, you will be able to:

- Add visualization items to reports.
- Choose an effective visualization.
- Format and configure visualizations.
- Import a custom visual.
- Add an R or Python visual.

Design a report layout

The page layout of the reports you create in Power BI Desktop will likely depend on the business requirements, the context of the underlying data, and the output requirements. For example, if you are designing a dashboard, you'll need to present high-level information on a single page. If you are designing a report, it is a multi-perspective view into your dataset, with visuals that represent different findings and insights from that dataset.

Report design best practice

The first step in designing a great report layout is choosing the correct format to use. It's likely that your manager, or whomever requested the report, will give you some requirements in terms of the format. If not, you'll need to imagine the audience and take a best guess at what kind of format they'll want.

For example, if your report users have a technical background and are looking for the nitty-gritty, you can use multiple, complex visuals that offer the most detail, along with interactive slicers. Conversely, if your users are just looking for quick data insights at a high level, you could use a small range of basic visuals.

Even if you have been given some layout requirements, you still need to carefully consider the audience of your report. Your goal is to provide the audience with the information they need, in the most optimal way. While you may have strong feelings towards what to data to display and how to display it, ultimately the report is not for you, it is for a dedicated audience that needs to make business decisions based on your report.

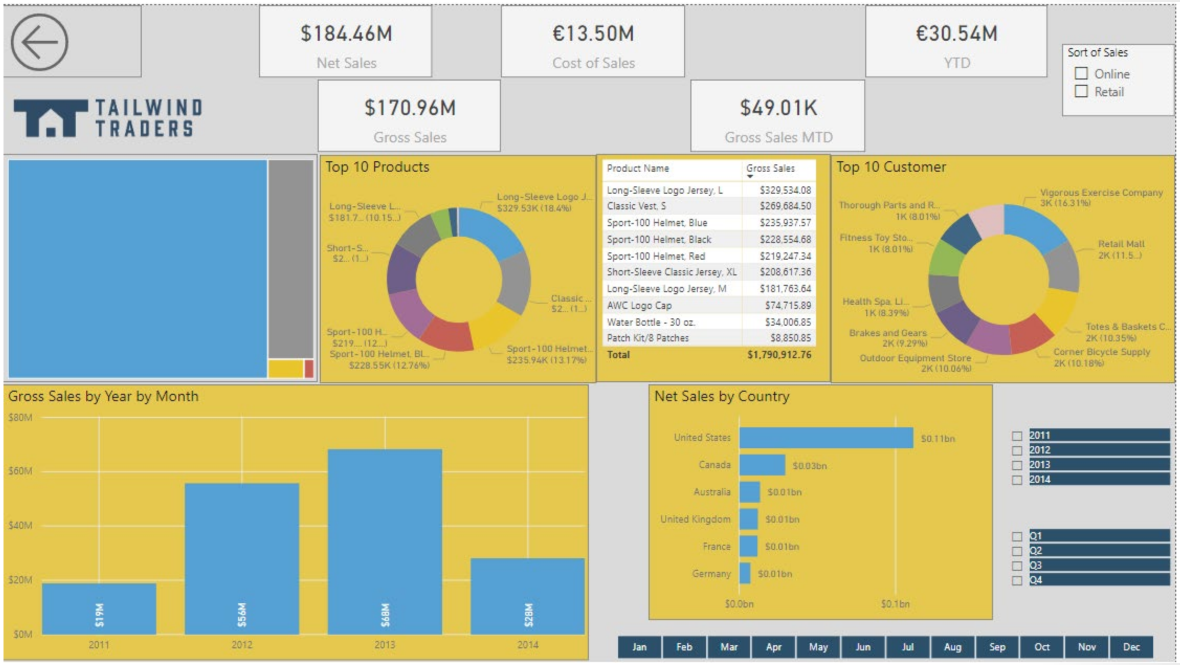
It is also important to consider the different needs that the end users within that audience may have. End users of your report might have hearing, motor, cognitive, or visual impairment. To cater for those needs, you'll have to create a report that offers an accessible experience, which means it is easy to navigate and understand by keyboard or screen reader users. You'll learn more about design and configuring your report for accessibility that later in this unit.

You also need to carefully consider each visual and element that you plan on using in the report. Everything should have a purpose, and you should consider how each element will look to your report users. While you might want to use lots of different types visuals for the sake of variety or to show off your skillset, sometimes a simple visual is all that you need. It's likely that your organization will have style guidelines for reports, in which case you'll have to adhere to particular color scheme and font. Do keep in mind also that the more visuals you use in your report, the more they impact on the performance of your report. You'll take a close look at visuals later in this unit.

Here are some other key guidelines for creating a well-designed report layout:

- Draw a sketch of your report layout, so you can get a quick picture of what will look like, before you spend lots of time physically designing it. You could even draw multiple sketches where you try out different ideas, then discuss these ideas with your team to help you choose the best layout design.
- Focus on the most important information. Highlight key parts of your report with a bright color or summary icon, so that it stands out, and users drawn to the most critical metrics.
- Select the right background for the context of your report. It is said that a white background makes your report look clean and business-like, a black background draws the eye to colorful highlights on the report, and an image used as a background adds numerous feeling.

The following image depicts an example of a badly designed layout; something you should avoid. At the end of this unit, you will see the same report but with an improved design.

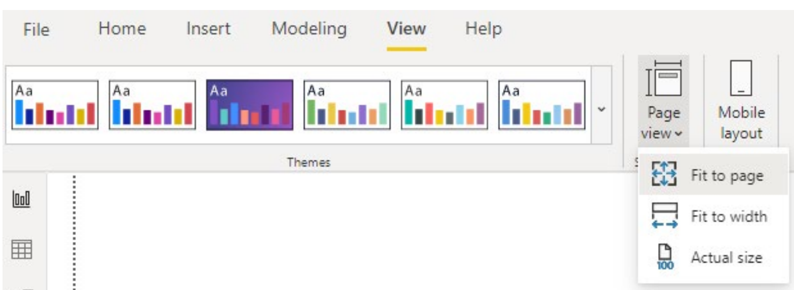


The following sections provide more detailed guidance for setting up the report page and using visuals.

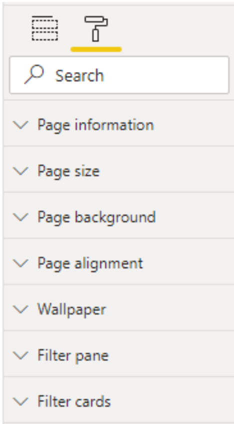
Report page

It is important to consider that you and the report users might view the reports on screens with different aspect ratios and sizes.

The default display view is **Fit to page**, which means that the contents are scaled to best fit the page. If you need to change this view, go to the **View** tab, select **Page view**, and then select your preferred page view option, as illustrated in the following screenshot.



To access the page settings, select the white space on your report canvas to open the **Format** pane. You can then configure the following settings to suit your needs: **Page information**, **Page alignment**, **Page size**, **Wallpaper**, **Page background**, and **Filter pane**.



Choosing effective visualizations

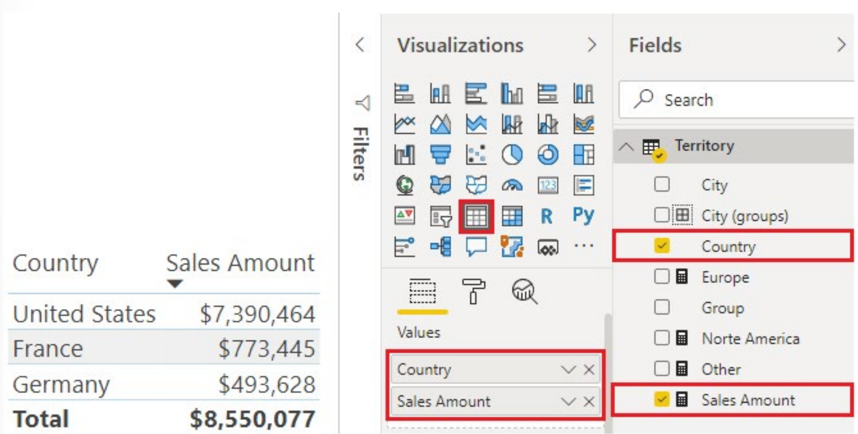
Power BI Desktop offers a range of out-of-the-box visualization options that are available directly from the **Visualizations** pane. When you select the fields that you want to display in a visualization, you can experiment with all the different visualization types to find the one that best suits your needs. If you can't find a visual that meets your needs, you can download other visuals from Microsoft AppSource or import your own custom visuals.

Depending on the type of data in your selected fields, one or more visualizations might not be suitable. For example, geographic data will not display well as a funnel chart or line chart visualization.

It is important that you choose an effective visualization to ensure that you display the data in the best way possible. The following sections outline the different types of visualizations that are available within Power BI Desktop, using the same data source for illustration purposes.

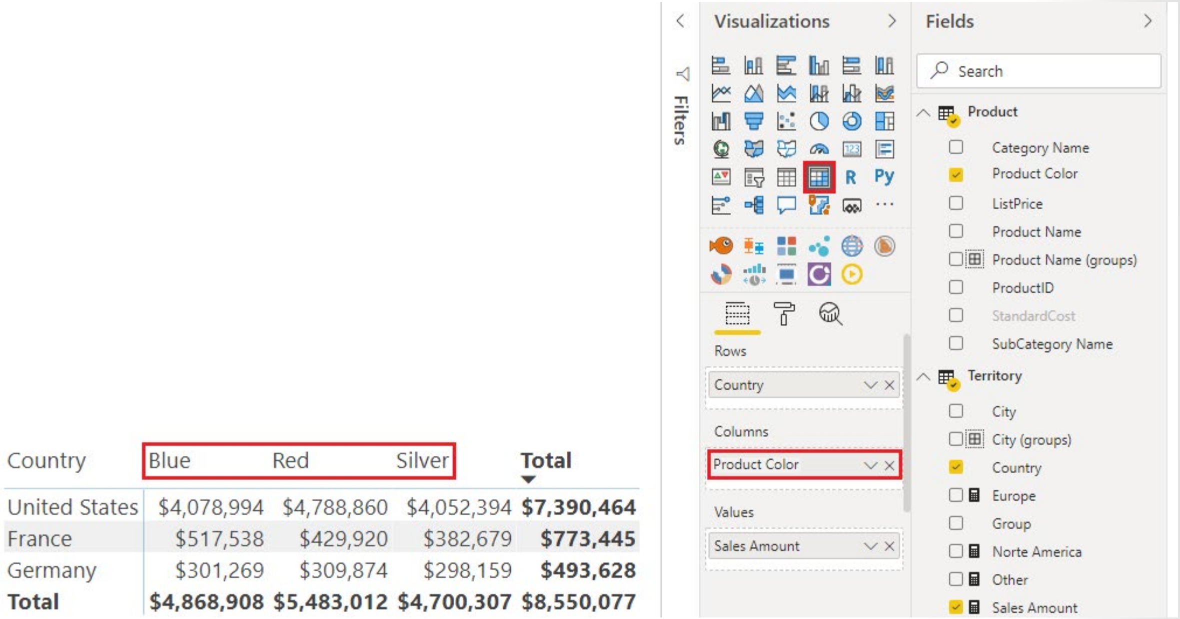
Table and Matrix visualizations

In the previous example, the **Table** visualization was selected by default. The table is a grid that contains related data in a logical series of rows and columns. The table supports two dimensions and the data is flat, which means that duplicate values are displayed and not aggregated. It can also contain headers and a row for totals.



The **Matrix** visualization looks similar to the table visualization; however, it allows you to select one or more elements (rows, columns, values) in the matrix to cross-highlight other visuals on the report page.

In the following image, notice that a new field called **ProductColor** was added to the columns, and the available colors are now spanning across the table, with the categories listed in rows.

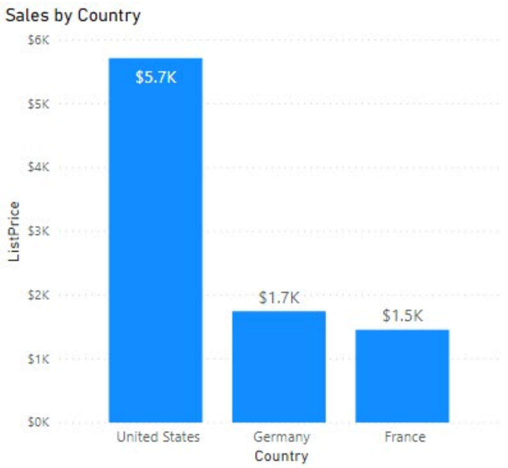


Country	Blue	Red	Silver	Total
United States	$4,078,994	$4,788,860	$4,052,394	**$7,390,464**
France	$517,538	$429,920	$382,679	**$773,445**
Germany	$301,269	$309,874	$298,159	**$493,628**
Total	**$4,868,908**	**$5,483,012**	**$4,700,307**	**$8,550,077**

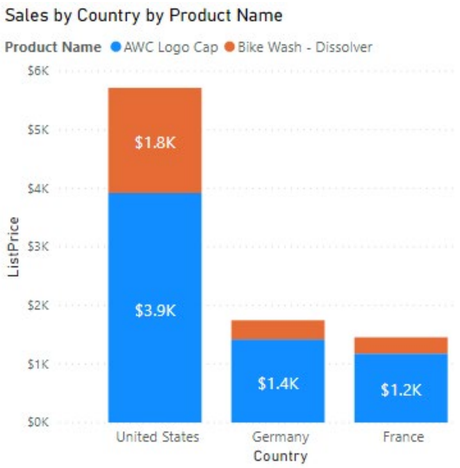
Bar and column charts

Power BI Desktop has a variety of bar and column chart visualizations that present specific data across different categories in a stacked or clustered format. The stacked format will stack the information items on top of each other.

For example, the following clustered column chart shows a single column with total sales for each country, whereas the stacked column chart shows data for sales by country, by product name. All sales data is stacked into one column to show you the total sales by country, broken down by how much each product contributed to the overall total sales.



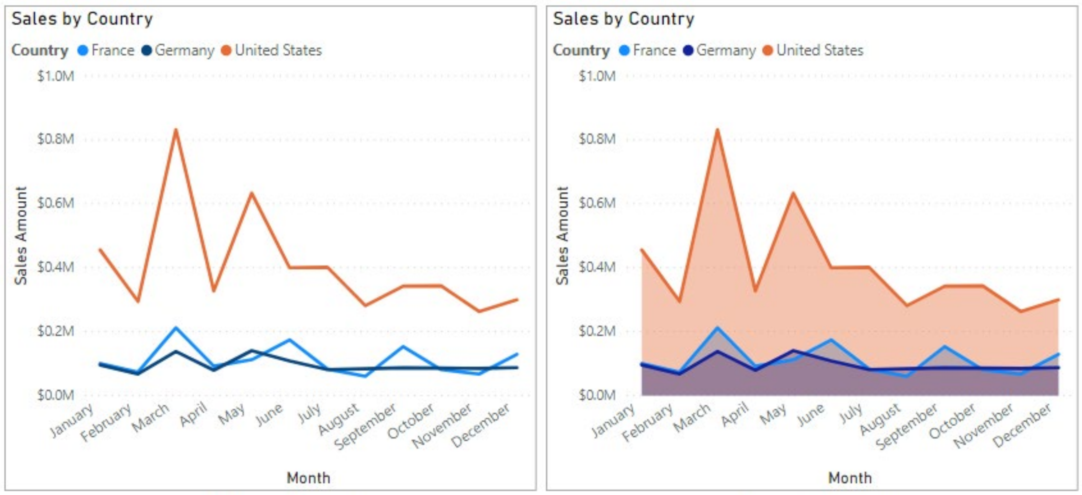
Clustered Column Chart



Stacked Column Chart

Line and area charts

The **line chart** and **area chart** visualizations are beneficial in helping you present trends over time. The basic area chart is based on the line chart, with the area between axis and line filled in. The main difference between these two chart types is that the area chart highlights the magnitude of change over time.

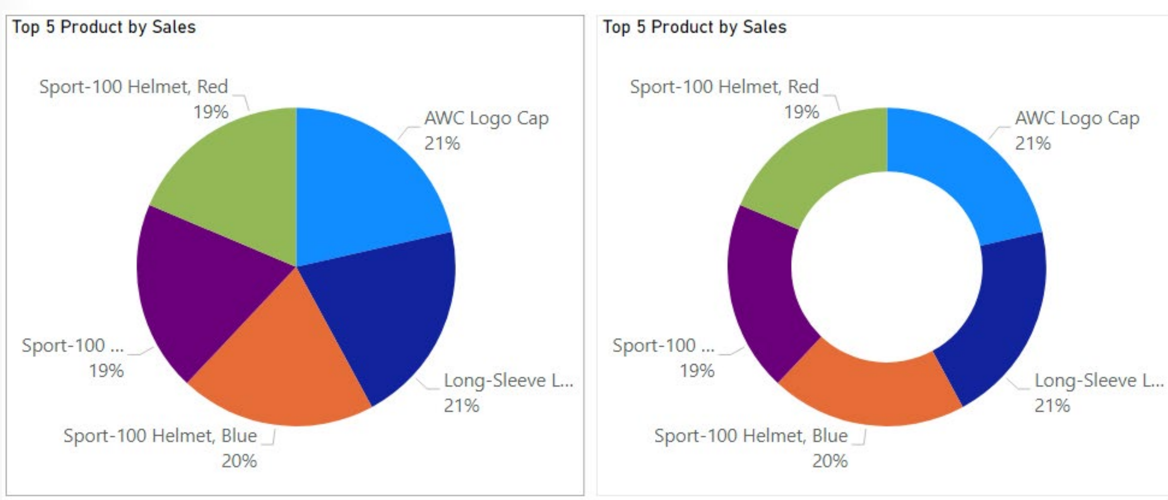


Line Chart **Area Chart**

Pie chart, donut chart, and Treemaps

The **pie chart**, **donut chart**, and **Treemap** visualizations show you the relationship of parts to the whole by dividing the data into segments. From a data analysis perspective, these charts are not useful because interpreting the data that they present can be difficult. However, these charts are often used for aesthetic reasons due to the colorful segments that they display. These charts are best suited for illustrating percentages, such as the top five sales by product or country, or any other available categories.

The pie chart is a solid circle, whereas the donut chart has a center that is blank and allows space for a label or icon.



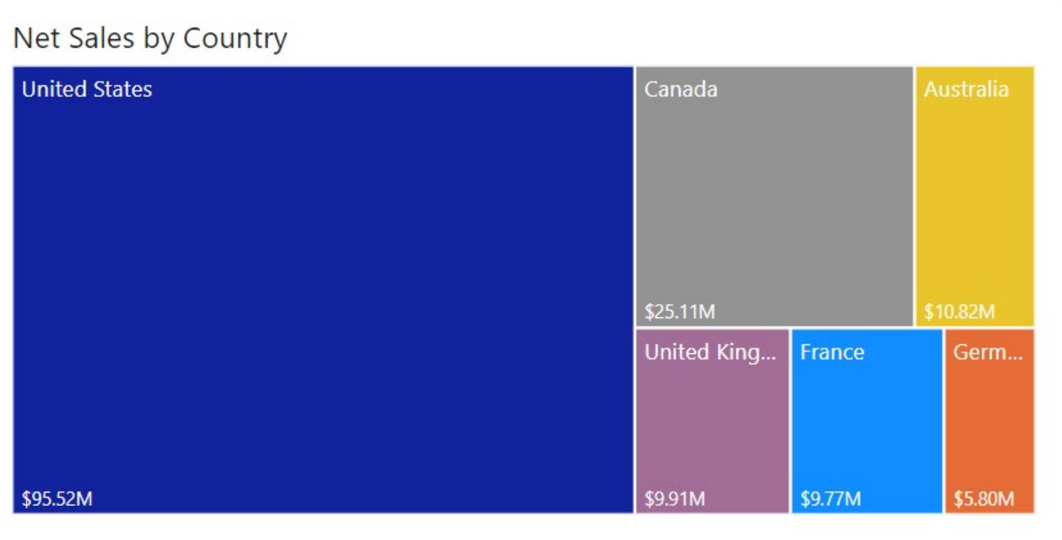
Pie Chart **Donut Chart**

When using pie charts, donut charts, and **Treemaps**, try to avoid presenting too many categories because it results in thin slices (or rectangles) that provide no added value to the user. If you do need to present all categories in your dataset, it's better to use another type of visual, such as a column chart.

Pie charts and donut charts present data by dividing it into slices, while the **Treemap** visualization displays data as a set of nested rectangles. Each level of the hierarchy is represented by a colored rectangle (branch) containing smaller rectangles (leaves). The space inside each rectangle is allocated based on the value that is being measured. The rectangles are arranged in size from top left (largest) to bottom right (smallest).

A **Treemap** is ideal to visualize:

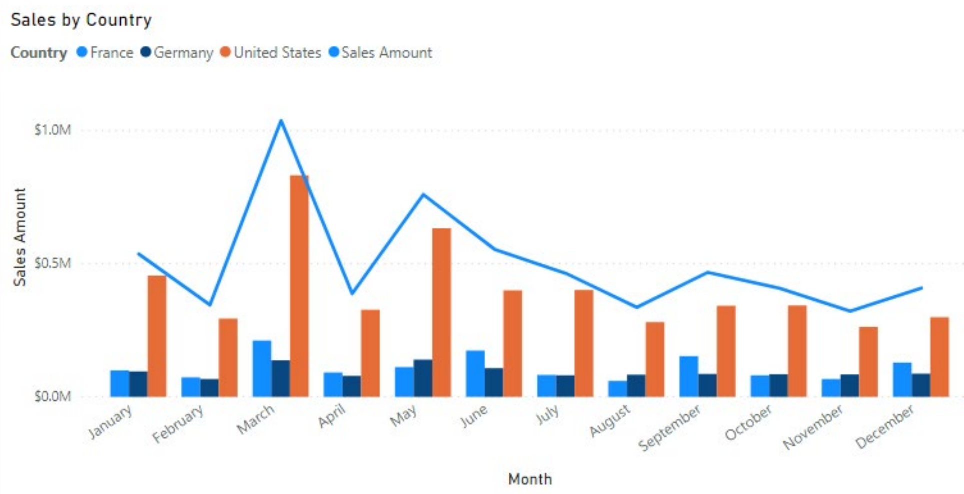
- Large amounts of hierarchical data when a bar chart can't effectively handle the large number of values.
- Proportions between each part and the whole.
- The distribution pattern of the measure across each level of categories in the hierarchy.
- Attributes, by using size and color coding.
- Spot patterns, outliers, most-important contributors, and exceptions.



Combo charts

The **combo** chart visualization is a combination of a column chart and a line chart that can have one or two Y axes. The combination of the two charts into one lets you:

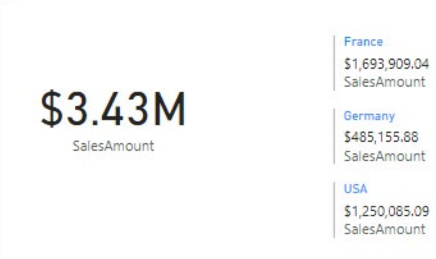
- Compare multiple measures with different value ranges.
- Illustrate the correlation between two measures in one visual.
- Identify whether one measure meets the target that is defined by another measure.
- Conserve space on your report page.



Card visualization

The **card** visualization displays a single value: a single data point. This type of visualization is deal for visualizing important statistics that you want to track on your Power BI dashboard or report, such as total value, YTD sales, or year-over-year change.

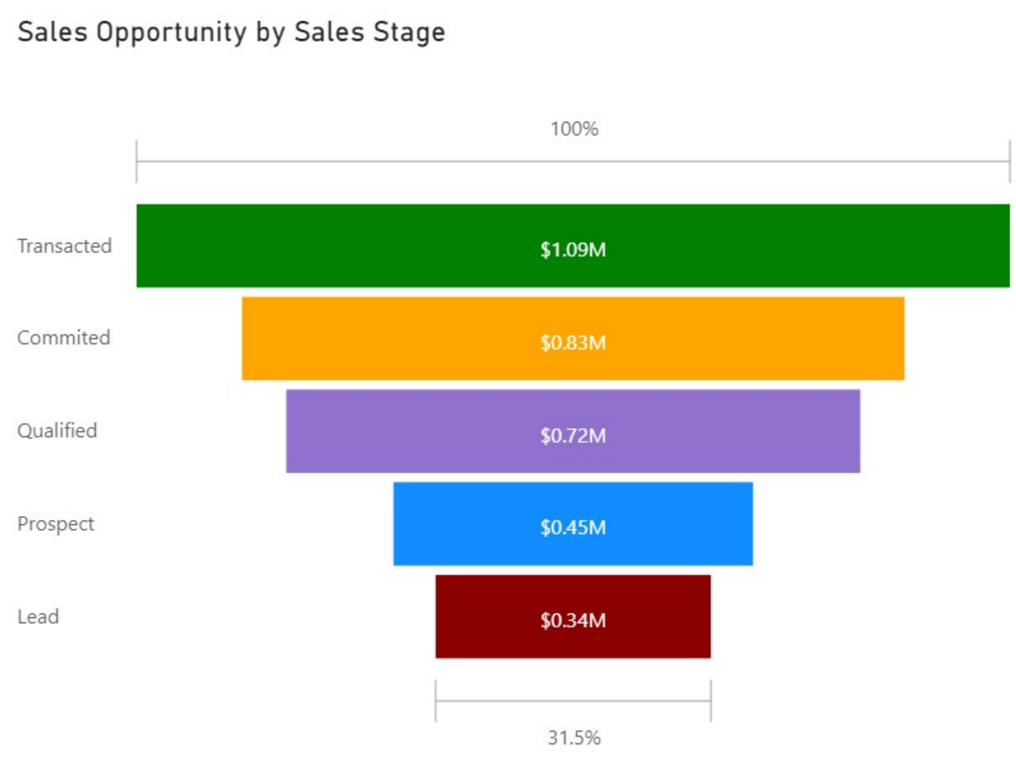
The **multi-row** card visualization displays one or more data points, with one data point for each row.



Funnel visualization

The **funnel** visualization displays a linear process that has sequential connected stages, where items flow sequentially from one stage to the next.

Funnel charts are most often seen in business or sales contexts. For example, they are useful for representing a workflow, such as moving from a sales lead to a prospect, through to a proposal and sale.



Funnel charts are great options in the following contexts:

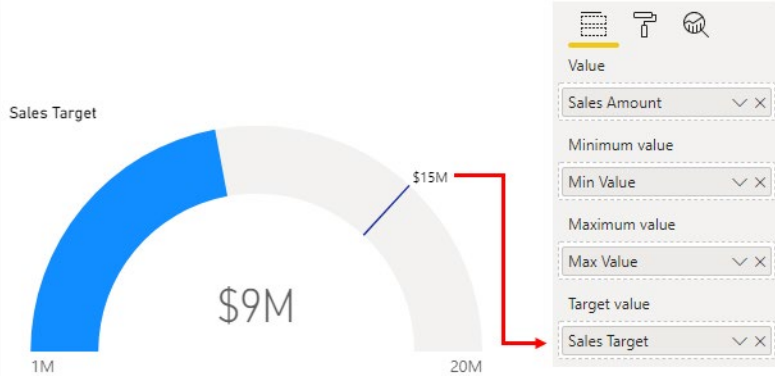
- When the data is sequential and moves through at least four stages.
- When the number of items in the first stage is expected to be greater than the number of items in the final stage.
- To calculate a potential outcome (revenue, sales, deals, and so on) by stages.
- To calculate and track conversion and retention rates.
- To reveal bottlenecks in a linear process.

Gauge chart

A radial gauge chart has a circular arc and displays a single value that measures progress toward a goal or target.

The value at the end of the arc represents the defaulted maximum value, which will always be double the actual value. To create a realistic visual, you should always specify each of the values. You can accomplish this task by dropping the correct field that contains an amount into the **Target value**, **Minimum value**, and **Maximum value** fields on the **Visualization** pane.

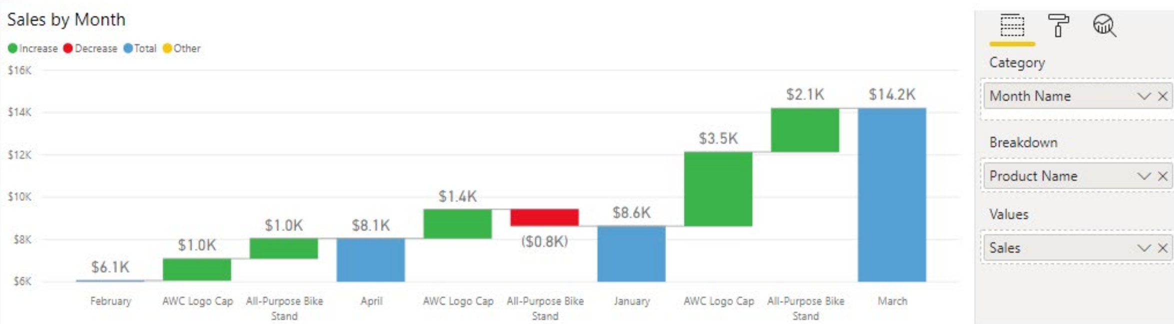
The shading in the arc represents the progress toward that target. The value inside the arc represents the progress value. Power BI spreads all possible values evenly along the arc, from the minimum (left-most value) to the maximum (right-most value).



Radial gauges can be used to show the progress that is being made toward a goal or target, or they can show the health of a single measure. However, radial gauges do take up a lot of space in comparison to the insights that they provide. It is more effective to use a pair of gauges with a spark line so users can see the trend and know what to do about it.

Waterfall visualization

The **waterfall** visualization (also known as a bridge chart) shows a running total as values are added or subtracted, which is useful in displaying a series of positive and negative changes. The chart consists of color-coded columns, so you can quickly identify increases and decreases. The initial and the final value columns often start on the horizontal axis, while the intermediate values are floating columns.



Waterfall charts can be used to:

- Visualize changes over time or across different categories.
- Audit the major changes that contribute to the total value.
- Plot your organization's annual profit by showing various sources of revenue to help determine the total profit (or loss).
- Illustrate the beginning and ending headcount for your organization in a year.
- Visualize how much money you earn and spend each month and the running balance for your account.

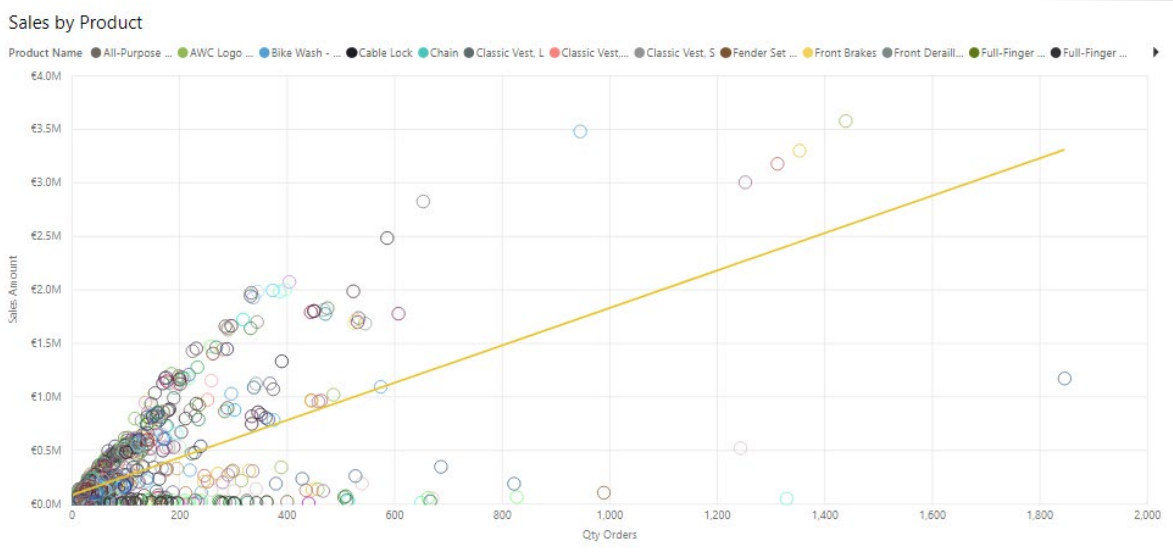
Scatter chart

The **scatter** chart visualization is effective when you are comparing large numbers of data points without regard to time. The scatter chart has two value axes to show: one set of numerical data along a horizontal axis and another set of numerical values along a vertical axis. The chart displays points at the intersection of an X and Y numerical value, combining these values into single data points. These data points might be distributed evenly or unevenly across the horizontal axis, depending on the data. You can set the number of data points, up to a maximum of 10,000.

You might want to use a scatter chart instead of a line chart because it allows you to change the scale of the horizontal axis. Scatter charts also allow you to:

- Show relationships between two numerical values.
- Plot two groups of numbers as one series of x and y coordinates.
- Turn the horizontal axis into a logarithmic scale.
- Display worksheet data that includes pairs or grouped sets of values.
- Show patterns in large sets of data, for example, by showing linear or non-linear trends, clusters, and outliers.
- Compare large numbers of data points without regard to time. The more data that you include in a scatter chart, the better the comparisons that you can make.

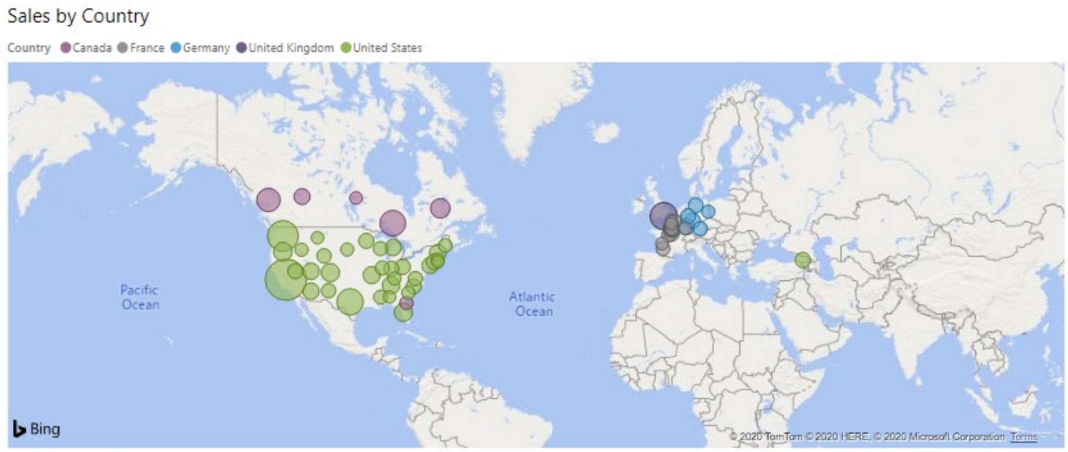
The following example shows a scatter chart that displays outliers (anomalies) with a trendline going up. The chart clearly shows that most products were sold at the same quantity, and only some products were sold in larger quantities. By identifying those outliers, you can run further analysis and break them down by country and region, which can help to improve logistics, decrease costs, and increase customer satisfaction.



Maps

Power BI integrates with Bing Maps to provide default map coordinates (a process called geocoding), so you can create maps. Together, they use algorithms to identify the correct location; however, sometimes, it's a best guess.

A *basic* map (*bubble* map) is used to associate categorical and quantitative information with spatial locations. This type of map visual displays precise geographical locations of data points on a map, as illustrated in the following image. A *fill* map uses shading, tinting, or patterns to display how a value differs in proportion across a geographical region. Similarly, *shape* maps use colors to display relative comparisons of geographical regions. You can also use an ArcGIS map to display graphical information in a more interactive way.



Slicer visualization

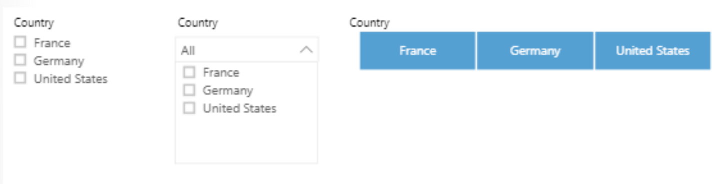
The **slicer** visualization is a standalone chart that can be used to filter the other visuals on the page. Slicers provide a more advanced and customized way of filtering, in comparison to the **Filters** pane, which is suited to more basic filtering operations. You can learn more about these two filtering options in another module.

Slicers come in many different formats, including list, drop-down, and buttons, and they can be formatted to allow the selection of only one, many, or all available values.

Slicers are ideal to:

- Visualize commonly used or important filters on the report canvas for easier access.
- Simplify your ability to see the current filtered state without having to open a drop-down list.
- Filter by columns that are unneeded and hidden in the data tables.
- Create more focused reports by putting slicers next to important visuals.

TIP: Using a slicer that is set to a drop-down format will defer the queries that are being sent to the dataset and can help improve performance.

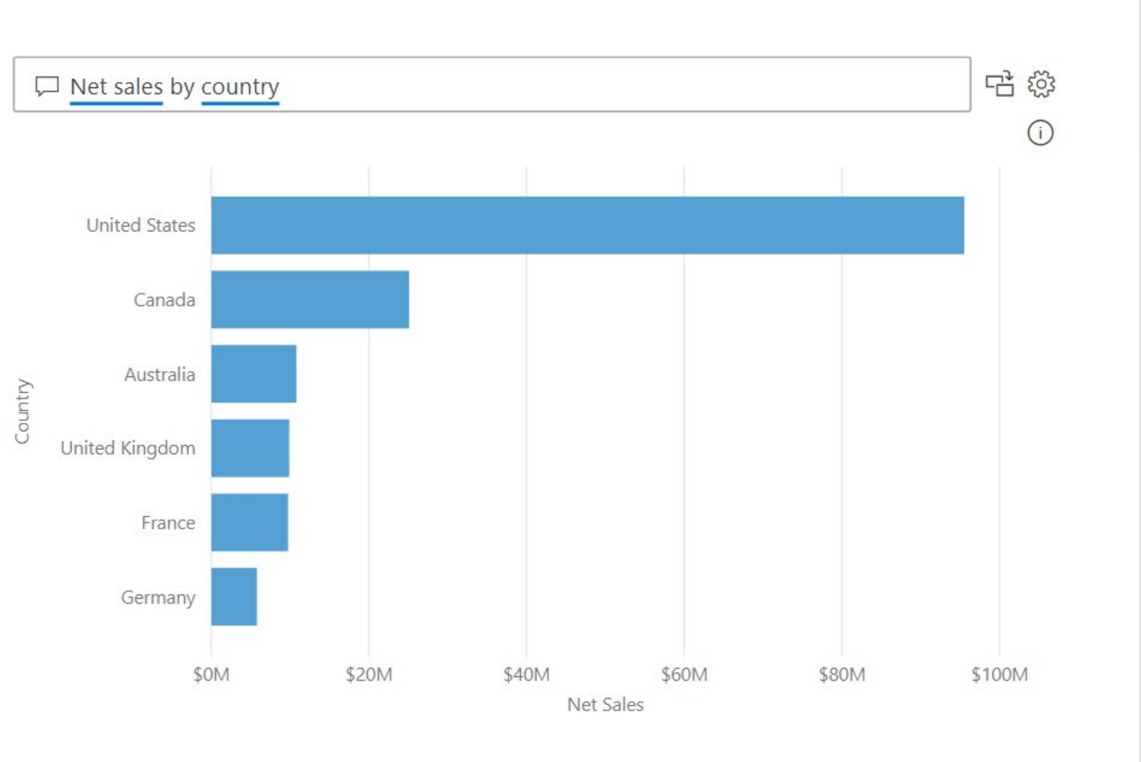


Q&A visualization

The **Q&A** visualization allows you to ask natural language questions and get answers in the form of a visual. This ability to ask questions is valuable to consumers and to you, the report author. This visualization type can help you create visuals in the report, and it can also be used as a tool for consumers to get answers quickly.

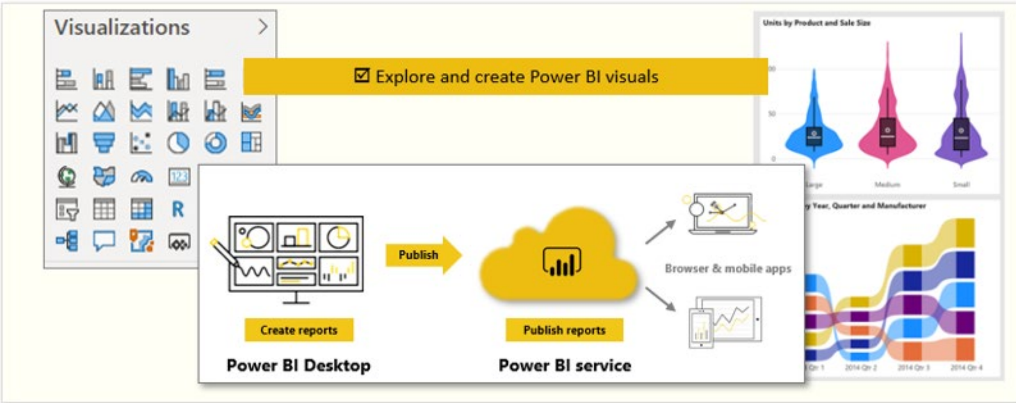
The Q&A visualization consists of the following four core components:

- The question box, where users enter their question and are shown suggestions to help them complete the question.
- A pre-populated list of suggested questions.
- An icon that users can select to convert the Q&A visual into a standard visual.
- An icon that users can select to open Q&A tooling, which allows designers to configure the underlying natural language engine. When entering natural language queries with Power BI Q&A, you can specify the visual type in your query. The following example illustrates how to implement **Net sales by country**.



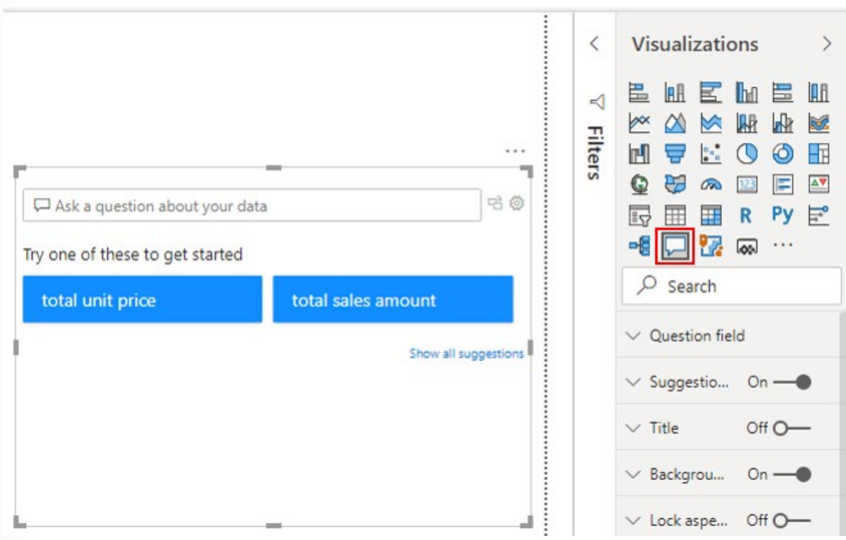
Adding visualizations to reports

Power BI has a variety of visuals that you can use to report on the data in your data model. Visuals allow you to present the important information and insights that you discovered in the data in a compelling and insightful way. The report consumers rely on these visualizations as a gateway to the underlying data.

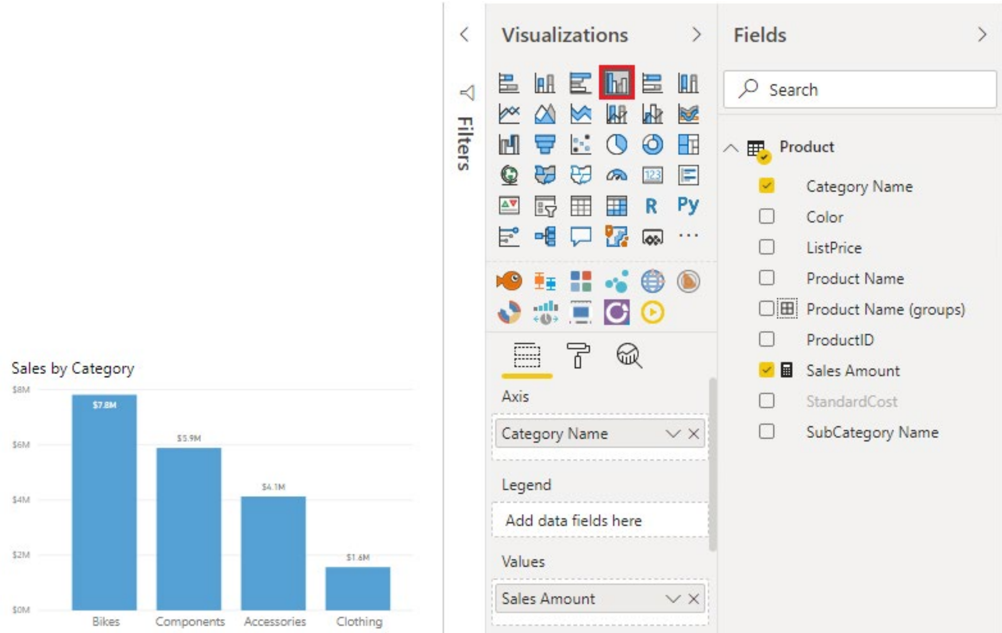


In Power BI Desktop, each visual is represented by an icon in the **Visualizations** pane. The types of visuals that are available include charts, maps, cards, a table, a matrix, and many more. You will learn how to select the correct visual later in this module.

In this example, you want to add a visualization to the report that displays sales data by category name. You start by selecting the **CategoryName** and **SalesAmount** fields in the **Fields** pane. Power BI Desktop then automatically selects a visualization for you, depending on the data type of the fields that you selected. In this case, the default visualization type is a table.



While the visual is selected, you can change the visualization type by selecting a different visual from the **Visualizations** pane.



Visuals

You might want to use a combination of visuals in your report, such as cards, charts, tables, slicers, and so on. It is important to use the right number of visuals on a page, and then size and position those visuals in the best way.

Number of visuals

Consider the number of visuals (including slicers) that you want to use on each report page. More visuals might have the opposite effect to what you are trying to achieve, your report might look too busy, and users might feel overwhelmed and not know where to look. Also, visuals are a key factor in the performance of your report, they contribute to performance issues. The fewer visuals you use, the better performance you'll get.

It's best to limit the number of visuals you use on a page. Examine each visual and ask yourself if it is necessary. If a visual does not add any value to the end user, you should not use it in your report.

Rather than using multiple visuals, you can provide information in other ways, such as drill through pages and report page tooltips. You'll learn more about these in subsequent units in this module.

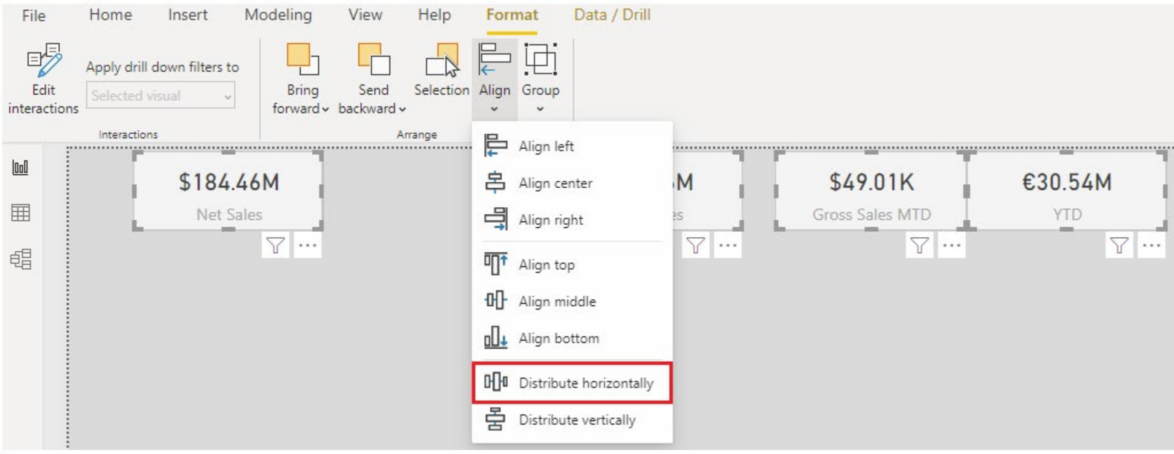
Position of visuals

When you add visualizations to a report, you can move those visuals to specific locations on the page, and make them bigger or smaller for a more effective display.

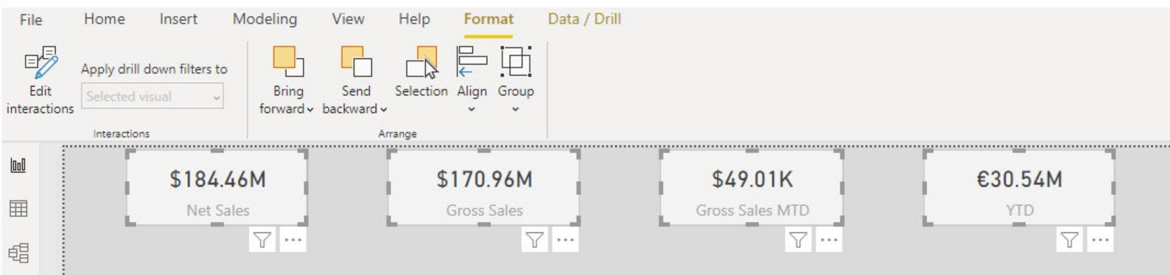
It is best practice to place the most important visual in the top-left corner of your report, as your report users will most likely from read left to right, and top to bottom. You might also want to place your organization's logo in or near this area. You can then arrange the other visuals accordingly.

To move a visualization, select any area of the visualization and then drag it to the new location.

To evenly distribute distance between visuals located on the canvas, you can also use the **Align** function. Use **CTRL**+click to select all of the visuals that you want to align, then select the **Format** tab and select **Distribute horizontally**.



The visuals will be evenly distributed against each other.



Size of visuals

When you add a visual to a report, Power BI determines the size of that visual by default. You can resize that visual to present the information it displays in the most optimal way. For example, if it is a small car visual, you might want to make it even smaller. Similarly, if it is a scatter chart visual with much data, you might want to make that visual a larger size, so users can see the data more clearly.

To resize a visual, select the visual to display its border, then and click and drag the dark frame handles to the size you want it to be.

Interaction of visuals

The visuals that you add to your report will interact with each other. For example, when you select an element in one visual, such as a product category, the other visuals will update in relation to that element, they might highlight or filter the specific data they display. Therefore, when you are designing the report, it is important to understand these interactions and consider how they might affect the overall user experience of the report. You have control over how interactions flow between the visuals, you might want to change a filter action to a highlight, and vice versa, or even prevent an interaction from happening. You'll learn how to do this later in this module.

Hierarchies in visuals

It is likely that you'll have a number of hierarchies in your data, so you should consider how these hierarchies will affect how the data displays in the visuals, and the navigation experience of your report users. You can set how hierarchies are presented in the visuals. You can also determine the hierarchical path of visuals, so you have full control over what level of detail can be accessed. You'll learn more about hierarchies later in this module.

Import a custom visual

In addition to the out-of-the-box visualizations in Power BI Desktop, hundreds of other developers have created a multitude of visuals for you to choose from. If you have a specific visual in mind, you can likely find it in the marketplace. If you can't find it, Power BI makes it possible for you to build your own.

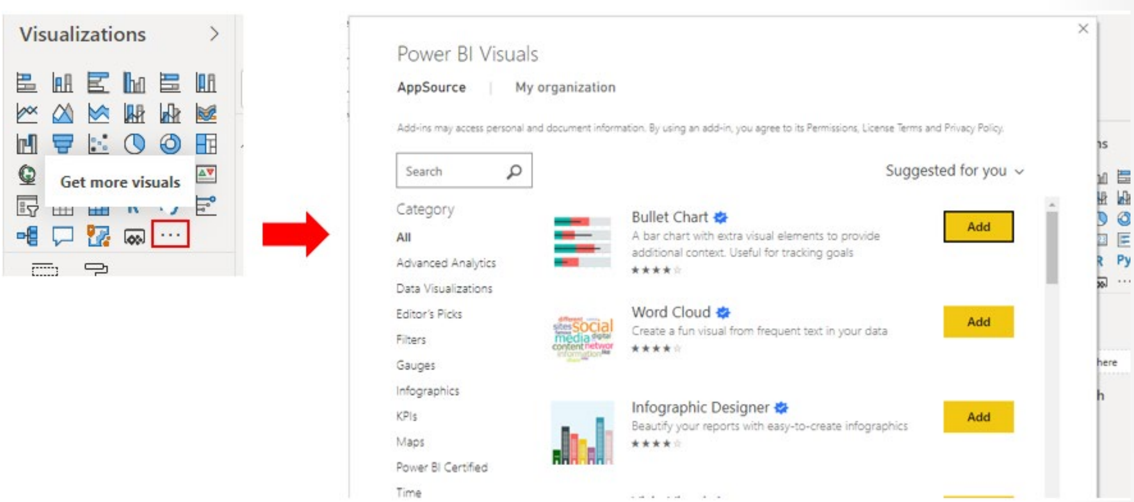
The custom visuals that are available in Microsoft AppSource are created by Microsoft and Microsoft partners. Some of these custom visuals are certified and some are not. The certified status means that the visual meets the Microsoft Power BI team code requirements; the visual is tested to verify that it doesn't access external services or resources and that it follows secure coding patterns and guidelines. The certification process is optional, so an uncertified visual is not necessarily unsafe to use.

NOTE: Some organizations prefer not to use custom visuals for security or other reasons. Before you import custom visuals, check with your organization to see whether custom visuals are allowed or not. If they are not allowed, you can still create reports in Power BI Desktop with them, but they will not render in Power BI service.

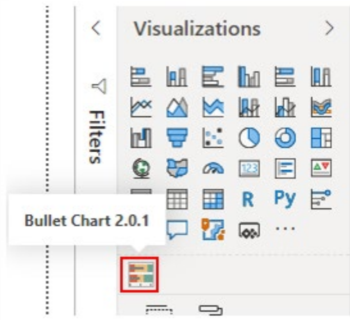
If you want to create your own custom visual, you can use the custom visual software development kit (SDK), which is an open-source tool based on NodeJS (JavaScript programming language) that is available on GitHub. The custom visual is packaged as a single Power BI Visual Tools (.pbiviz) file that you can import into Power BI Desktop.

Creating a custom visual is beyond the scope of this unit, so in this example, you will import a custom visual from AppSource.

In the **Visualizations** pane, select the **Get more visuals** icon and then select **Get more visuals**. On the window that displays, locate and select the visual that you want to import and then select **Add**.



The new visual will appear under the other visuals in the **Visualizations** pane. To add the visual to your report, select its icon. You can then add fields to the visual and customize its formatting, just like you would for any other visual.

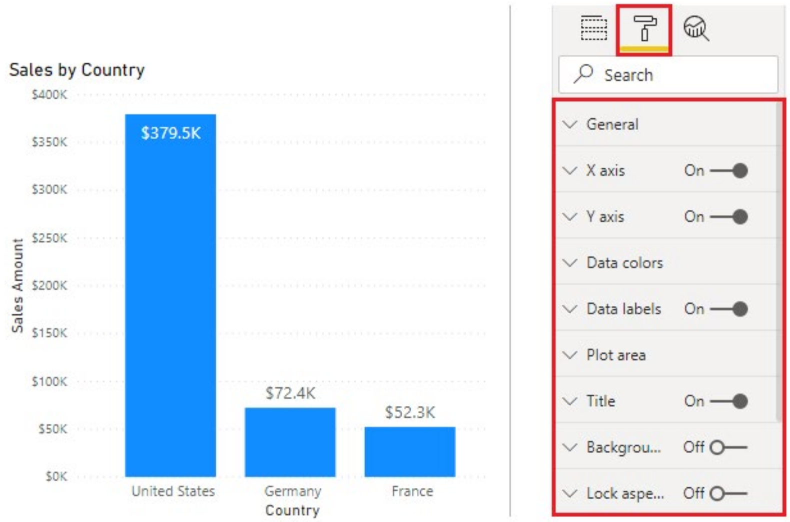


Format and configure visualizations

Power BI Desktop gives you a variety of options for customizing how your selected visualizations look, such as the colors and format of the text that they contain. You should take time to explore the options to determine what impact they each have on a visual.

In this example, you will format and configure the default clustered column chart visualization to better meet the needs of your report requirements.

Start by selecting the visualization on the canvas, and then select the **Format** button (paint roller icon) to display the **Format** pane.



The formatting options that are available will depend on the type of visualization that you selected.

Common formatting options include the **Title**, **Background**, and **Border**. In the **Title** section, you can add a title to the visual, if it does not have one, or edit the title, if it has one already. The aim of the title is to clearly describe what data is being presented in the visual. You can format the title by changing the text, text size, font, color, background, and alignment. The subsequent section shows an example of customizing a title.

In the **Background** section, you can set any color or image as the background for the visual. If you plan to use an image as a background, try to select an image that won't have lines or shapes that would make it difficult for the user to read the data. It is best to keep a white background so the presented data can be clearly seen. The subsequent section shows an example of customizing a background.

In the **Border** section, you can set a border around the visual to isolate the visual from other elements on the canvas, which helps make it easier for the user to read and understand the data. You can change the border color and radius to be consistent with your color scheme.

If a **General** section is available, you'll be able to set the precise size and place for your visual on your canvas. This option might be suitable if the drag-and-drop feature is not placing the visual exactly where you want it to be. It can also be useful to ensure that you have aligned specific visuals consistently.

You might also be able to format the colors and labels for specific data values. In the **Data colors** section, you can set the colors that you want to use for the data values in the visual. You can use different colors for different fields, but always try to be consistent when it comes to selecting those colors. It is best to use the same color scheme throughout the report. In the **Data labels** section, you can change fonts, size, and colors for all labels in the visual. Try to use solid colors so the labels are clearly visible. For example, if the background is white, use a black or dark grey color to display your labels.

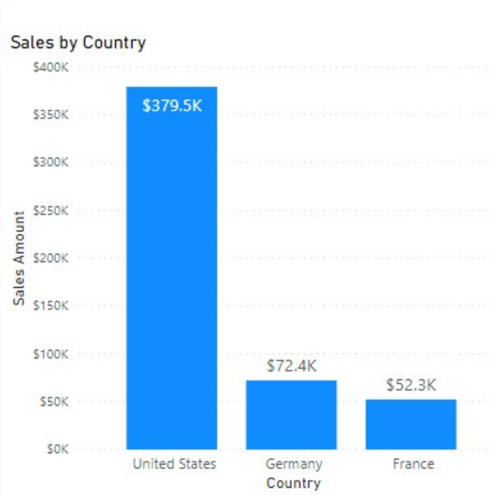
The **Tooltips** section allows you to add a customized tooltip that appears when you hover over the visual, based on report pages that you create in Power BI Desktop. Tooltips is a great feature because it provides more contextual information and detail to data points on a visual. The default tooltip displays the data point's value and category, but your custom tooltips can include visuals, images, and any other collection of items that you create in the report page. The subsequent section shows an example of customizing a tooltip.

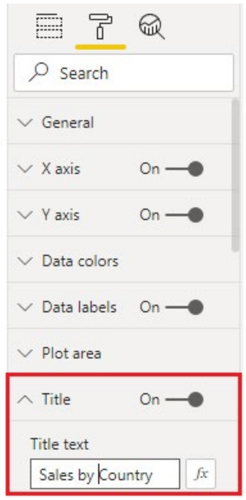
As you make changes in the **Format** pane, notice that the visualization updates immediately to reflect those changes. If you need to revert the changes that you make, select the **Revert to default** option at the bottom of each section in the **Format** pane.

In the following examples, you will edit the title, change the background, and add a tooltip.

Title

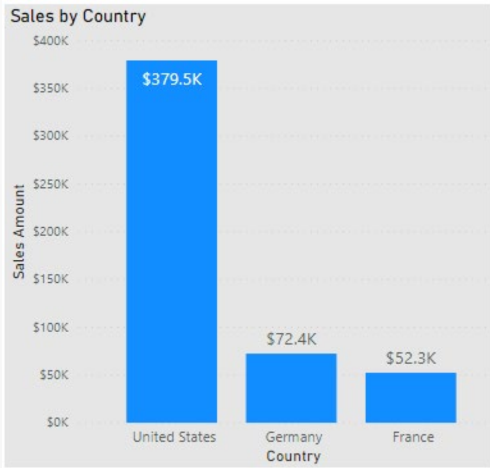
You can edit a default title and add a title, if you don't have one. In this example, you will select the column chart visualization and then, in the **Format** pane, scroll down and expand the **Title** section. Edit the current title by adding a space between **Sales** and **Amount**, and then increase the font size to 16 points.

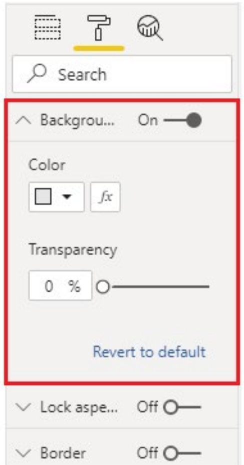




Background

It is best practice to keep the default white background so the presented data can be clearly seen. However, you can change the default background color to make a visualization more colorful and easier to read or to match a particular color scheme. In this example, continue with the column chart that is selected and then, in the **Format** pane, expand the **Background** section and change the color to light grey.



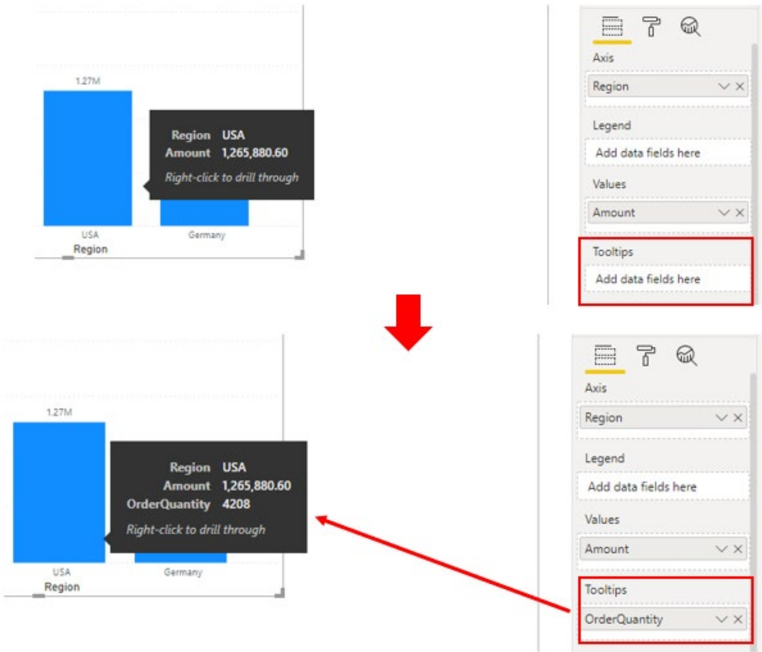


Tooltip

Using tooltips is a clever way of providing more contextual information and detail to data points on a visual. When you add a visual, the default tooltip displays the data point's value and category, but you can customize this information to suit your needs. For example, you might want to provide your report users with additional context and information, or specify additional data points that you want users to see when they hover over the visual.

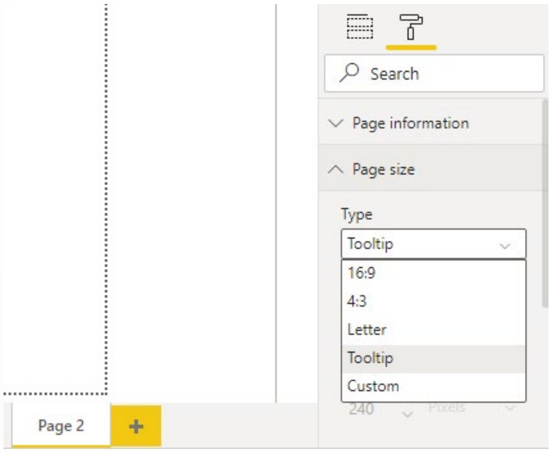
To expand on the data points that are displayed in the default tooltip, you can drag a field (value) from the **Fields** panel into the **Tooltips** bucket. However, you should not add many more fields to the tooltips because adding too many fields can introduce performance issues and slow down your visuals.

The following image shows the default tooltip first and then the customized tooltip that displays additional data.

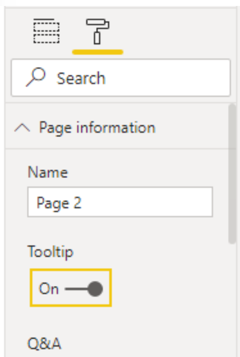


Another way to use tooltips is to display graphical information. The process of adding this type of tooltip is not as straightforward, but it is worthwhile. You would begin by creating a new page in the report.

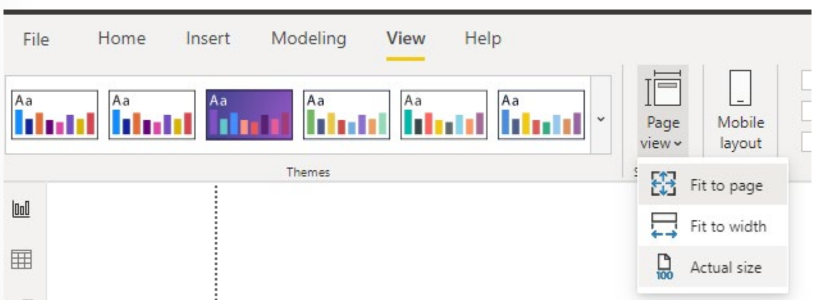
Open the new page and then open the **Format** pane. Expand the **Page Size** section and then select **Tooltip** from the **Type** list.



In the **Page information** section, turn the **Tooltip** slider to **On** so that Power BI registers this page as a tooltip page

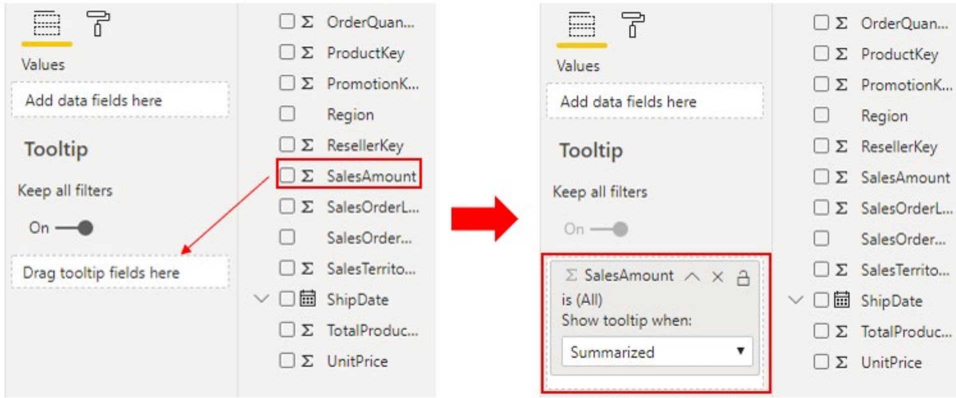


Tooltips have limited canvas space, so to ensure that your visuals appear in the tooltip, on the **View** tab, set the **Page view** option to **Actual size**.



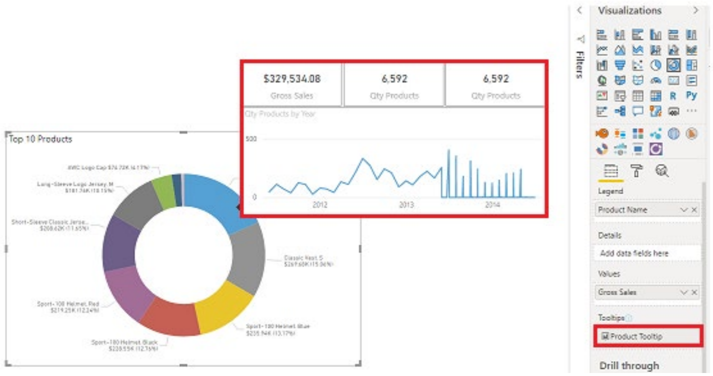
Next, add one or more visuals to the tooltip page, in the same way that you would on any other report page.

Now, you need to specify the fields for which you want the tooltip to display. Select the tooltip page and then select the **Values** tab in the **Visualizations** pane. Drag the fields from the **Fields** pane into the **Tooltip** bucket. In this example, you will drag the **SalesAmount** field into the **Tooltip** bucket.



Return to the report page and apply the tooltip to one or more visuals on that page. Select a visual and then, in the **Format** pane, scroll down to the **Tooltip** section. Turn the tooltip option **On** and then select your tooltip page from the **Page** list.

When you hover over the visual, the tooltip will display.



Basic Interactions

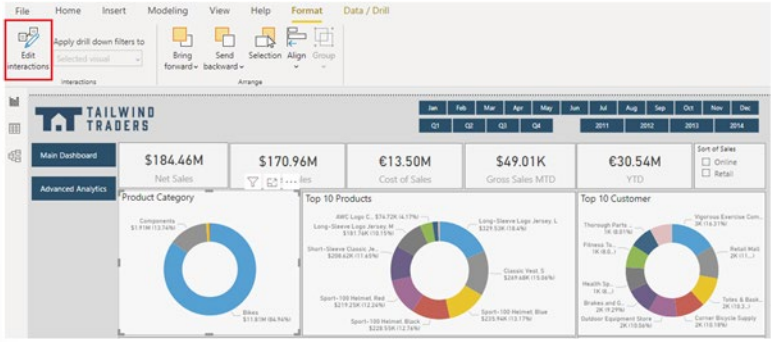
In Power BI Desktop your report is dynamic. When you make a selection on one visual in the report, other visuals might change to reflect that selection. Similarly, if there are hierarchies in your data, you can move up and down the hierarchy to see the data at different levels.

In this unit you'll learn about how the basic interactions work, and also how to use hierarchies in your visuals. In the next unit, you'll learn how to edit the interactions, and use the drill-through features.

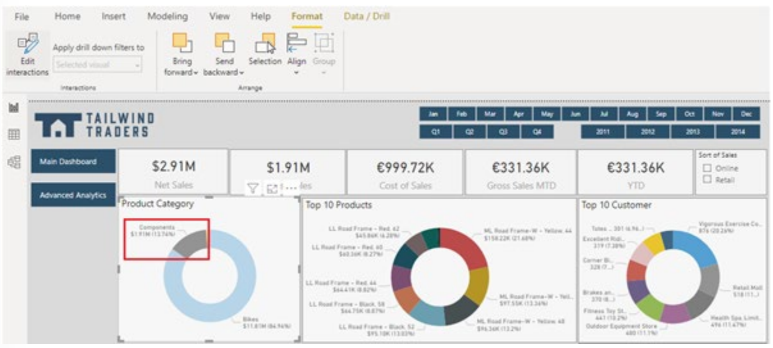
View interactions

When you have multiple visuals on the same report page, they all interact with each other, so you should become familiar with these interactions and see how your report changes.

Compare the following two images. In the first image, the data displays at a high level.



When you select an element in a visual, such as **Components** in the **Product Category** visual, the other visuals update to reflect your selection, as illustrated in the second image.

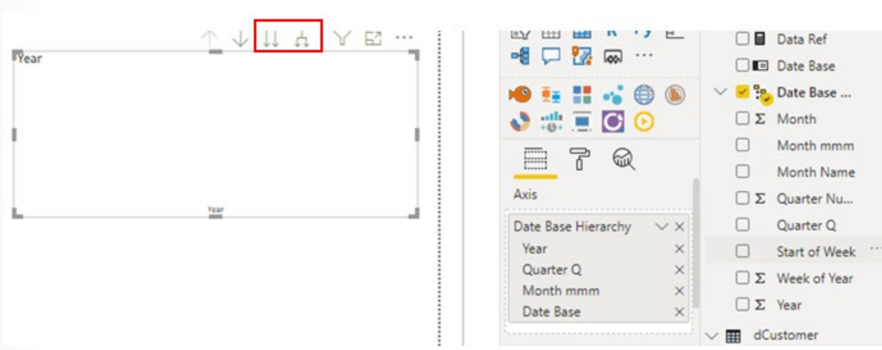


When you have become familiar with the interactions, you might want to make changes in order to control how those interactions flow between the visuals. You'll learn how to do this in the next unit.

Use hierarchies

A hierarchy is a structure in which groups are ranked one above the other, according to a specific status. Think of your own organizational hierarchy, where you have the CEO at the top level, then managers in the middle level, and employees at the lower level. Similarly, you'll likely have hierarchies for your data in Power BI. For example, you could have a time hierarchy, with levels such as year, quarter, month and day, or a product hierarchy, with levels such as category, subcategory and product.

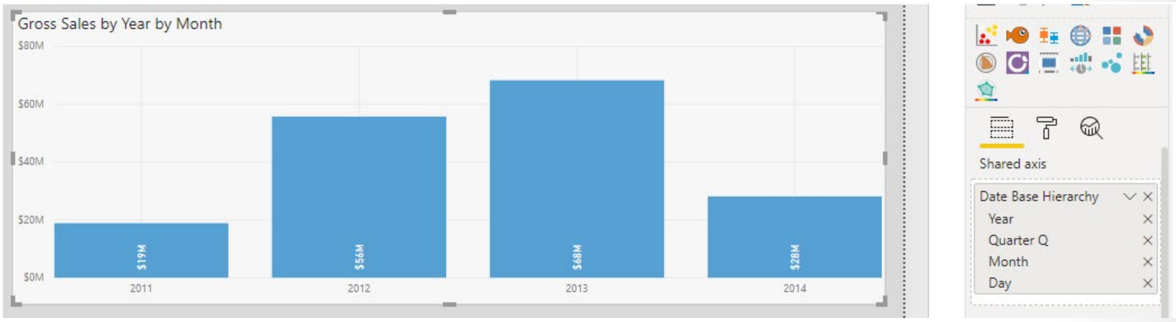
In regards to dates, Power BI creates hierarchies for you automatically. When you select the hierarchy in the **Fields** pane, the date hierarchy is added to the **Axis** field well on the **Visualizations** pane, and a blank visual is created for you, ready for additional fields. You'll also see the hierarchy icons above the visual, as illustrated in the following image.



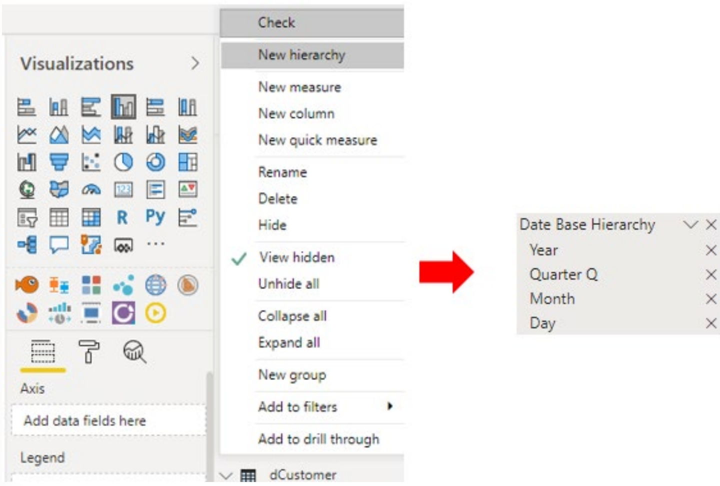
When you add another field to the visual, the visual becomes useful, and you can then use the hierarchy icons to navigate through the hierarchy - Power BI creates a predefined drill path for the data. In the following image, the **Gross Sales** field was added to the visual and you can see the data at the highest level (year), then when the **Expand all down one level in the hierarchy** button was selected, the hierarchy was expanded down by one level (quarter). If you were to select the button again, the visual would update to display the next lower level in the hierarchy (month), and so on.

Another navigation option is the **Go the next level in the hierarchy** button. When you select either hierarchy option, select the **Drill** up button to move back up hierarchy.

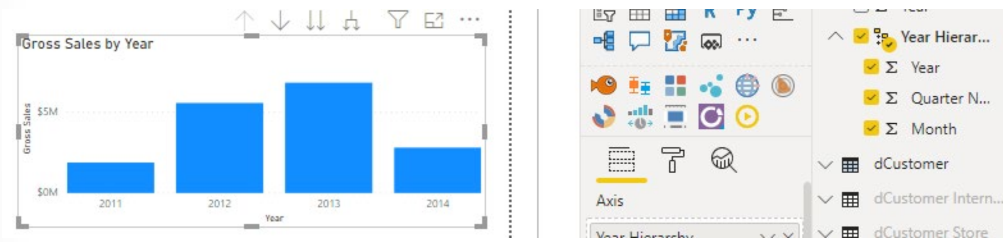
As you can see, the default date hierarchy feature in Power BI is quick and easy to use. However, if you have your own date table, you can use that to create the hierarchy instead. In the following image, the **Year**, **Month**, **Quarter** and **Week of Year** fields were added to the **Axis** field well on the **Visualizations** pane. The data result is the same as with the default hierarchy, and you can navigate through the hierarchy in the same way.



You can also predefine the hierarchy path for your report users, and remove the guess work from them. For example, you might want t prevent them from viewing a particular hierarchy level. To do so, remove all of the fields from the **Axis** well, then in the **Fields** pane, right-click the field you want to set as the top level of the hierarchy, and then select **New hierarchy**. The new hierarchy displays ion the list in the **Fields** pane. You can now drag and drop other fields into the new hierarchy, or right-click each field and select **Add to hierarchy**.



Ensure your visual is selected, then select the new hierarchy field in the **Fields** pane to apply it to the visual.



Once again, you can use the hierarchy buttons above the visual to expand down the hierarchy and drill back up again.

Configure conditional formatting

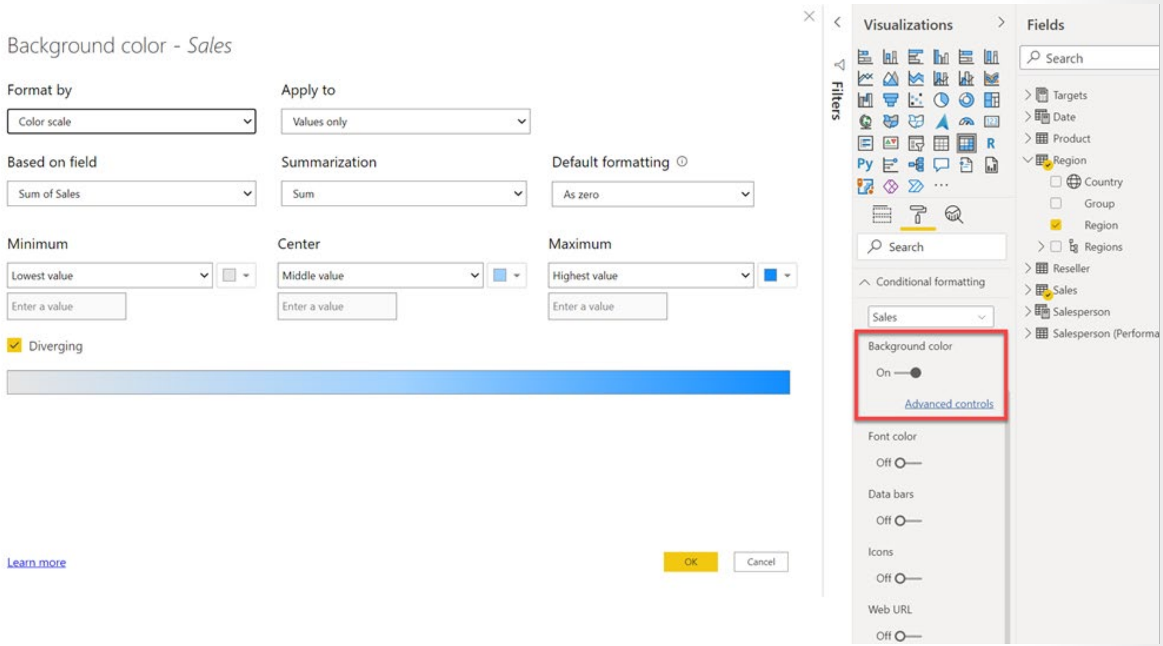
Conditional formatting in Power BI Desktop allows you to specify customized cell colors, including color gradients, based on field values, or represent cell values with data bars or KPI icons, or as active web links.

You might want to use conditional formatting to highlight or differentiate the data that is displayed in your visual. This will allow you and other users to see key data insights at a glance.

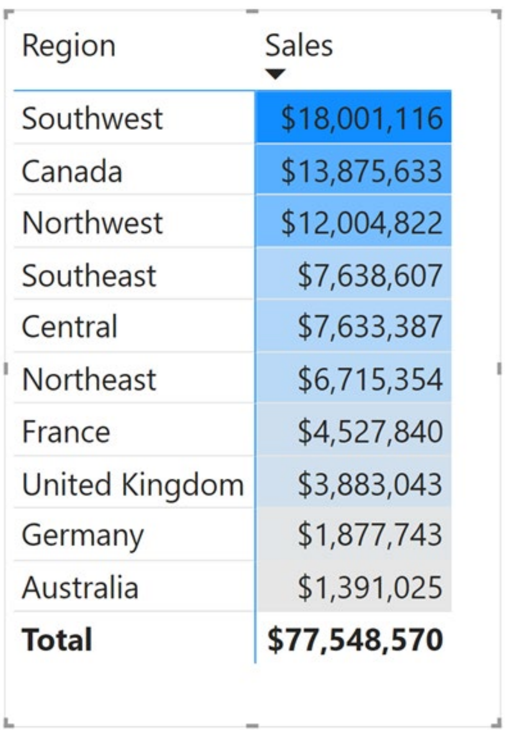
For example, you could set up conditional formatting for your sales figures. If the sales amount falls below zero, you could display this in red, a color that is associated with danger, so users will see this clearly and know that they need to take action. Conversely, you could set a value for your sales target, then display amounts over that target amount in a green color, to signify that target is met and all is going well.

Conditional formatting is available in Power BI. Here we see it used in two visuals: **Table** and **Matrix**, where it is possible to set different conditions on a column. You can apply conditional formatting to any text or data field but the formatting needs to be based on a field that has numeric, color name or hex code, or web URL values.

In this example, you select the table visualization, then in the **Format** pane, expand the **Conditional formatting** section. Turn the background color on, then select the **Advanced controls** option. In the window that displays, set a condition to change background color of the cell to red for cells with low values and green background to cells with high values.



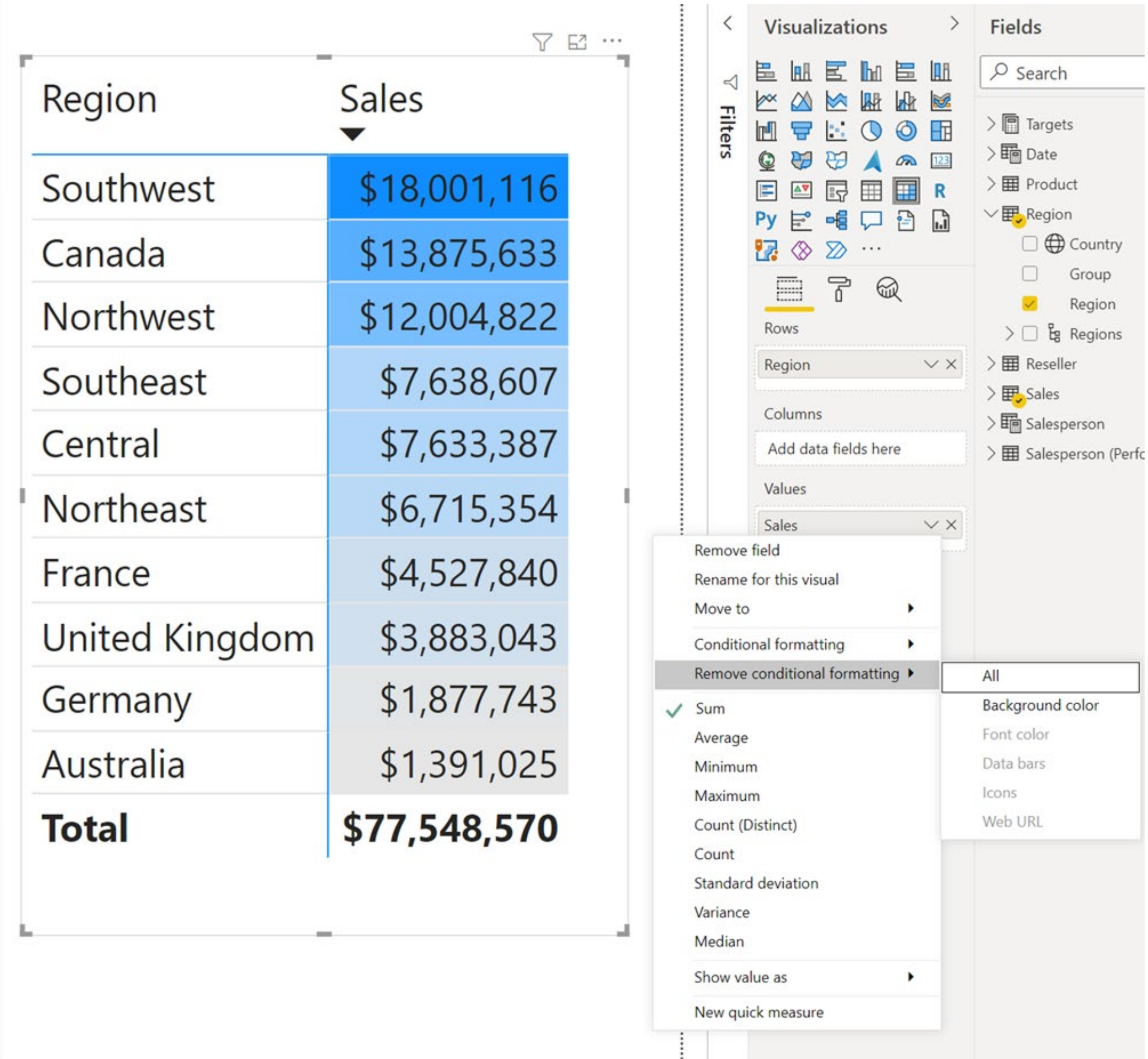
The Power BI conditioning function will automatically detect highest and the lowest number in each column and apply background coloring according to the values.



Region	Sales
Southwest	$18,001,116
Canada	$13,875,633
Northwest	$12,004,822
Southeast	$7,638,607
Central	$7,633,387
Northeast	$6,715,354
France	$4,527,840
United Kingdom	$3,883,043
Germany	$1,877,743
Australia	$1,391,025
Total	**$77,548,570**

If you want to remove the conditional formatting that you set, select the **Values** tab on the **Visualizations** pane and right-click the value (field) that you set the formatting against. Select **Remove condition-**

al formatting, then select the type of formatting you want to remove, for example **All** or **Background color**.



Design report navigation

Report navigation is the way in which your report users move from one page in your report to the next, move from one visual to another, and go back to where they started from. The design of your report navigation is very important because if users cannot easily find their way around your reports, they will become frustration and have a negative experience.

You can use a range of buttons and bookmarks when designing your report navigation, and you can further enhance this navigation experience with the use of conditional formatting.

Add navigation buttons

To design the navigation within your report, you can create a new Navigation page in your report, and add navigation buttons there. You can also use a combination of both options. When users click on one of these buttons, they are brought directly to a different page within the report, which you can hide, so it can be only be accessed through the Navigation page buttons.

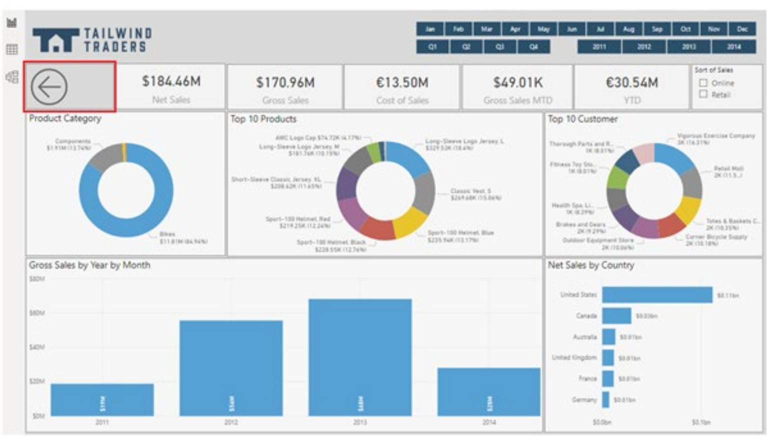
In this example, you create a Navigation page on which you add two navigation buttons to your other pages.

Start by adding a button, as you did in the previous unit. This time, when you expand the **Actions** section in the **Visualizations** pane, select **Page navigation** as the action type, and then select the page in your report that is the **Destination** for the button.

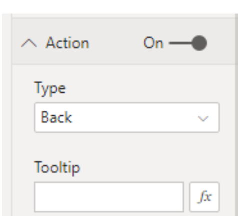
When you have set up the first navigation button, copy and paste that button to create the second navigation button, so you preserve the formatting you applied to the first button. Then change the title and destination for the second button.

Now when you select a button, you are brought directly to the assigned page destination. When you are on that destination page, to return to the Navigation page, you can use a **Back** button. Here are two back navigation options:

- Select **Back** button from the main **Buttons** menu, then reposition the button to where you want it to sit on the page.

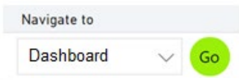


- Select **Blank** button from the main **Buttons** menu, reposition and customize the button as required, then select **Back** as the action type.



Conditionally set the navigation destination

You can use conditional formatting to set the navigation destination based on the output of a measure. One reason you might want to use this type of navigation method is to save space in your report. For example, rather than using multiple navigation buttons (as illustrated in the previous image), you can use a single button to navigate to different pages based on the user's selection (as illustrated in the following image).

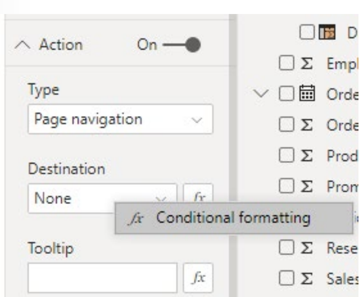


Other reasons for using this type of navigation include:

- To specify the logical path that your report users should take. In other words, you determine the order in which users view each page.
- To tell a data-driven story. For example, you could use it to give your employees a message that is backed up by the data. This could be useful to help drive change, such as increase sales.
- To create a reporting portal where users can navigate to a set of reports.

To use conditional formatting to set the navigation, start by creating a single-column table that has the names of the navigation destinations. In the table, ensure that the entered values match your report page names.

When you load the table, add it to the report page as a single-select slicer. Next, add a page navigation button. In the **Actions** section, ensure **None** is set as the **Destination**, then right-click the destination and select **Conditional formatting**.



On the **Destination** window, select the name of the column you created. Then you'll see that based on the user's selection the button can navigate to different pages. Configure the conditional formatting to complete your navigation design.

Lesson Review

If your organization wants to transform the way that it presents data to management and stakeholders, and wants to move from a text and tabular report format to a more visual format, you can use Power BI visuals. These visuals help report users get quicker, easier access to the information that they need to make their business decisions, from a report that is more visually pleasing.

To help your report audience connect and interact with these visuals, you can create a combination of visuals in your Power BI Desktop and then customize those visuals to ensure that they comply with organizational requirements.

Power BI Desktop offers a range of visual options that you can quickly add to your report, and it also gives you the ability to import custom visuals from a rich library that you can use to solve additional business problems. You can select the most effective visuals for your report needs and take advantage of Power BI's formatting options to customize those visuals to meet your organization's requirements.

If Power BI Desktop didn't provide visuals as the means to effectively communicate the insights that you find in your data, your user might find it difficult to access the information that they need and might ultimately struggle to make good business decisions.

Now that you have added visuals to your report, your managers can access the underlying data. The added bonus of a cleverly-designed report in Power BI that contains a variety of visuals is that your managers and all other report users will enjoy the experience of using your report.

Visuals allow you to better communicate information that is hidden in raw data. Visuals can highlight obscure trends in ways that tabular data might not. Power BI includes many compelling visuals that you can use to illuminate your data and tell better stories.

Continue your journey

Want to learn more about Power BI visuals? Check out these resources:

- **Radial gauge charts in Power BI**[1]
- **Change how visuals interact in a Power BI report**[2]
- **How-To: Display 2-letter country data on a Power BI map**[3]
- **Tips and tricks for color formatting in Power BI**[4]
- **Tutorial: Adding formatting options to a Power BI custom visual**[5]

1 https://docs.microsoft.com/power-bi/visuals/power-bi-visualization-radial-gauge-charts/
2 https://docs.microsoft.com/power-bi/service-reports-visual-interactions/
3 https://go.microsoft.com/fwlink/?linkid=2101354&clcid=0x409
4 https://docs.microsoft.com/power-bi/visuals/service-tips-and-tricks-for-color-formatting/
5 https://docs.microsoft.com/power-bi/developer/custom-visual-develop-tutorial-format-options/

- **Best design practices for reports and visuals**[6]
- **Design effective reports in Power BI**[7]

Knowledge Check

Question 1

What is the benefit of using a report tooltip?

☐ To give users the ability to export data from the visual.

☐ To provide additional detail that is specific to the context of the data that is being hovered over.

☐ To give users additional information about a report visual, such as the author and date/time it was created.

Question 2

Do you need to import custom visuals each time you want to use them when you are developing a new report?

☐ Yes, custom visuals must be imported from AppSource each time you start developing a new report unless you pin the custom visual to the visualizations pane.

☐ No, custom visuals are always available for selection under the Visualization pane.

☐ No, custom visuals only need to be imported once and will always remain in Power BI for future use in a new report.

6 https://docs.microsoft.com/power-bi/visuals/power-bi-visualization-best-practices/
7 https://docs.microsoft.com/en-us/learn/paths/power-bi-effective/

Enhance the Report

Introduction

Organizations use reports to monitor and record performance and identify trends and variances. Organizations rely on the information provided by reports when making decisions. Reports drive organizational behavior and action, at every level, in every aspect.

After you've added visuals to your report, you can make further improvements and fine tune the report before finally sharing it with the report audience. In this module, you'll apply the functions available in Power BI Desktop's report editor to your visuals, to transform your report into a data driven story. You'll provide users with a more effective report layout and navigation experience, and give them additional tools, so they can dive deeper into the information you present in your visuals. You'll then publish the report, so users can access the information they need to make decisions.

Imagine you work as a Power BI Developer for Tailwind Traders. Your managers are finding it difficult to make good business decisions based on the current quality of the reports. Your managers request one report that displays all of the information they need to carry out their yearly planning and forecasting activities, and ultimately make better strategic organizational decisions. They want the report to be concise, accurate, and well-designed, and it must display information in an interesting way that is easy to navigate and complies with current accessibility standards.

You have already made a start on the report by adding and customizing visuals. Now you need to take the report to the next level to meet management's requirements.

By the end of this module you will be able to:

- Apply slicing, filtering, and sorting
- Performance tune reports
- Comment on reports
- Use advanced interactions and drillthrough
- Add buttons, bookmarks, and selections
- Use Key Performance Indicators
- Publish and export reports

Apply slicing, filtering, and sorting

Power BI Desktop provides three tools that you can use to edit and configure interactions between the visualizations you add to your report: slicers, filters and sorting.

The process of filtering allows you to remove all of the data you do not need, so you can focus on the data that you do need. You can apply filtering directly using the **Filters** pane, or by adding and using a slicer. Slicers and filters are similar, they both let you filter out the unnecessary data. You should try out both options to see which one is the best mechanism for your report situation. You might decide to use one option over the other, or use a combination of both.

Contrary to filtering, the process of sorting allows you to highlight the important information without removing any of the data.

Add a slicer

A slicer is a type of filter that you can add to your report, so users can segment the data in the report by a specific value, such as by year or geographical location. The slicer narrows the portion of the dataset that is shown in the other visualizations in the report.

You might want to use a slicer to:

- Provide quicker access to commonly used or important filters.
- Make it easier to see the current filtered state without having to open a drop-down list.
- Filter by columns that are unneeded and hidden in the data tables.
- Create more focused reports (by putting slicers next to important visuals).
- Defer queries to the data model by using a dropdown slicer, particularly when using DirectQuery.

Slicers are not supported for input fields and drilldown functions.

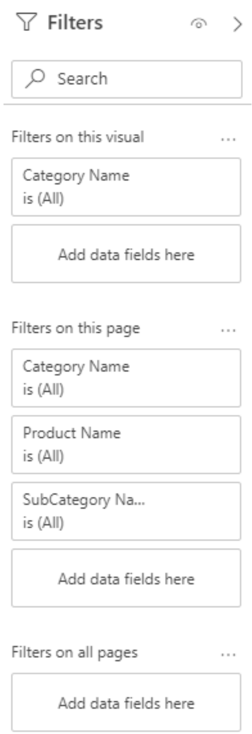
When you add a slicer, you can change that slicer to populate a list of items that you want to use to filter the elements of your page, and you can make that list appear in a dropdown format, if you want to save space for more important data on your report page. Rather than using a list format, you can turn your slicer into buttons, to make it easier for end-users to filter data. You can also use your slicer with date type columns, so you can select a different data range using the slider.

To apply a slicer, select the **Slicer** icon in the **Visualizations** pane. Then in the **Fields** pane, select the fields you want to include in the slicer, or drag them into the slicer visualization.

The visualization turns into a list of items (filters) with check boxes that you can use to segment the data. When you select the box of an item, Power BI will filter (slice) all of the other visualizations on the same report page, as illustrated in the following image.

You can add as many slicers as you want to a report page. If you use a list type of slicer, you can configure the selection controls. Select the slicer, then in the **Format** pane, expand the **Selection controls** section to view the options:

- **Single select** - This option is **Off** by default. It ensures only one item can be selected at a time.
- **Multi-select with CTRL** - This option is **On** by default. It allows you to select multiple items by holding down the **Ctrl** key.
- **Show "Select all"** - This option is **Off** by default. If you turn this option on, a **Select all** check box is added to the slicer. You might want to add this option so you can quickly select or deselect all of the items in the list. If you select all items, selecting an item deselects it, allowing an is-not type of filter.



While slicers are very useful, if you want to filter your data in a basic way, you do not need to add slicers to your report. Power BI Desktop has a **Filter** pane that can handle the basic slicer operations. So, depending on your requirements, you might save time and effort by avoiding the use of slicers and simply using the **Filter** pane instead. This has the added benefit of reducing the total number of visuals on a report, which will improve performance.

Customize the filters

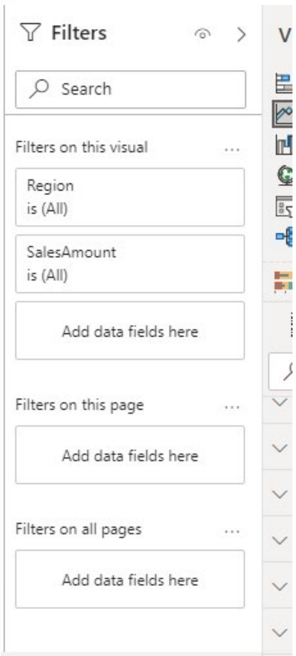
From the report user perspective, the **Filters** pane contains filters that you, as the report designer, have added to the report. The filters allow users to interact with the visuals at the report, the page, and the visual level.

As a report designer, you can customize the **Filters** pane in Power BI Desktop as follows:

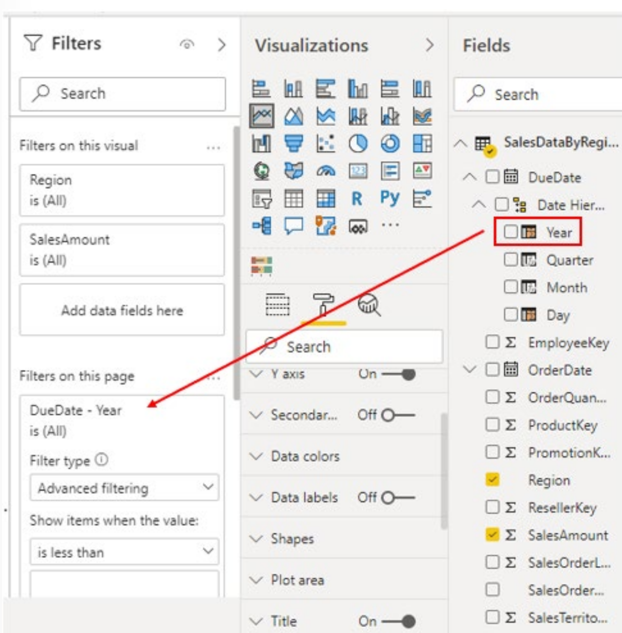
- Add and remove fields to filter on.
- Change the filter state.
- Format and customize the **Filters** pane so that it feels part of your report.
- Define whether the **Filters** pane is open or collapsed by default when a consumer opens the report.
- Hide the entire **Filters** pane or specific filters that you don't want report consumers to see.
- Control and even bookmark the visibility, open, and collapsed state of the **Filters** pane.
- Lock filters that you don't want consumers to edit.

You can expand and collapse the **Filters** pane, so you can hide it when you do not need it. When you expand the **Filters** pane, depending on the item in the report that you have selected, you will see the following sections:

- **Filters on this visual** - Filters that apply to the selected visual and nothing else. This section only displays if you have a visual selected.
- **Filter on this page** - Filters that apply to the whole page you have currently open.
- **Filter on all pages** - Filters that apply to all of the pages in your report.
- **Drillthrough** - filters that apply to a single entity in a report



To apply a filter, drag and drop a field from the **Fields** pane into the relevant section of the **Filter** pane.



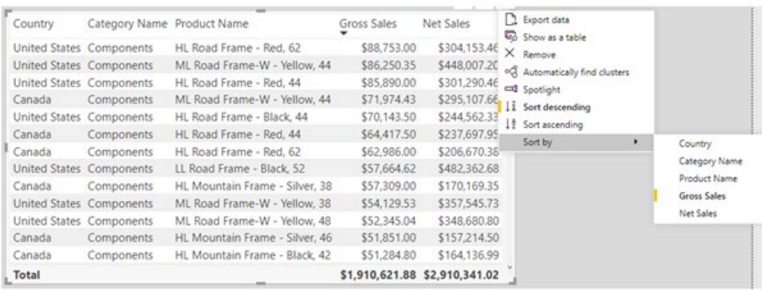
Sort data

You can sort the data displayed in your visuals, so they display exactly how you want them to.

Sorting helps you to display the most important data in the most logical way, such as in alphabetical or numeric order. This basic task can help you to make big business decisions. For example, if you display products with the highest sales first, you help the end-user to see what product is the most popular among the customer base. Similarly, the products with low sales can be discontinued or replaced with new products, in order to increase revenue.

To sort a visual, start by selecting the **More options** (...) button in the upper-right corner of the visual. You have three sorting options:

- **Sort descending** - Sorts the visual by the selected column in the order of greatest value to smallest value.
- **Sort Ascending** - Sorts the visual by the selected column in the order of smallest value to greatest value.
- **Sort by** - Sorts the data by a specific column. Hover over this option to display the list of columns that you can select from.



)

Performance tuning reports

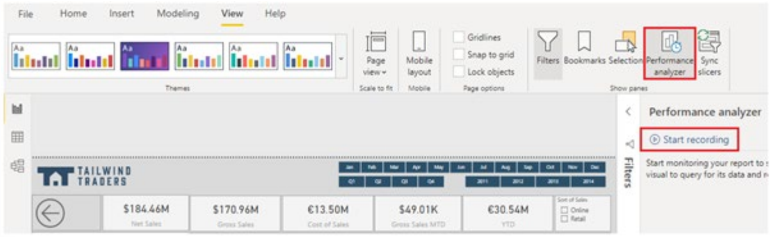
When you have completed creating your report, the performance of that report depends on how quickly data can load onto the report page. You should test out your report in the Power BI Report Server, to see how it works from an end-user perspective. If you experience any issues yourself, or if the report users have reported issues, you need to investigate the cause of those issues, and take measures to tune the report for more optimized performance.

Analyze performance

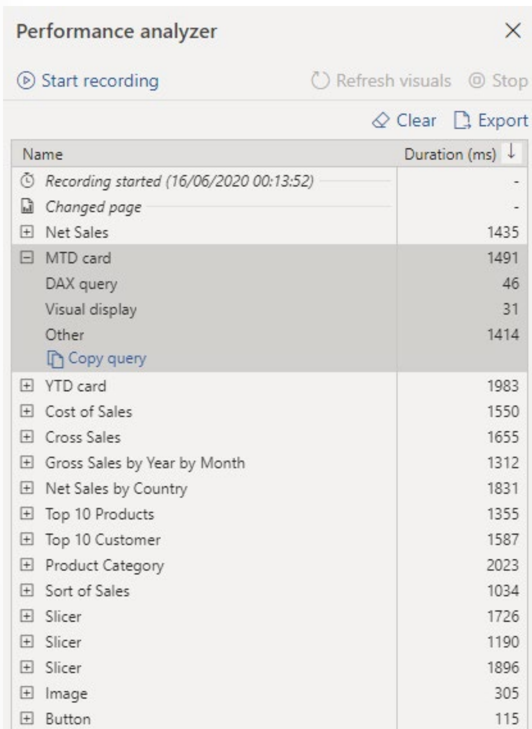
To investigate the cause of issues, your first port of call is the **Performance analyzer** tool within Power BI Desktop. **Performance analyzer** allows you to find out how each of your report elements, such as visuals and DAX formulas, are performing. **Performance analyzer** provides you with logs which measure (in time duration) how each of your report elements performs when users interact with them. By looking closely at the durations in the logs, you can identify which elements of the report are the most (or least) resource intensive. You can find where the bottlenecks are and this gives you a good starting point for making changes.

Before you run **Performance analyzer**, ensure you clear the visual cache and data engine cache, otherwise the results will not be accurate. Also, you should set up the report so that it opens on a blank page.

When you have cleared the caches, and opened the report on the blank page, to run the **Performance analyzer**, go to **View** tab, select **Performance analyzer**, and then select **Start Recording**.



Interact with your report as you would expect a user to, then stop the recording. You will see the results of your interactions displaying in the **Performance analyzer** pane as you work. When you are finished, select the **Stop** button. You can then analyze the results in the **Performance analyzer** pane. You will see the performance results of each item in the report, in milliseconds, under the **Duration** column. In the following image, you can see that all of the items on the report take less than two seconds to load, which is not bad. You can expand an item in the list to view more detailed information and identify what exactly is causing the issue, such as the Data Analysis Expressions (DAX) query, the visual display or something else (other).

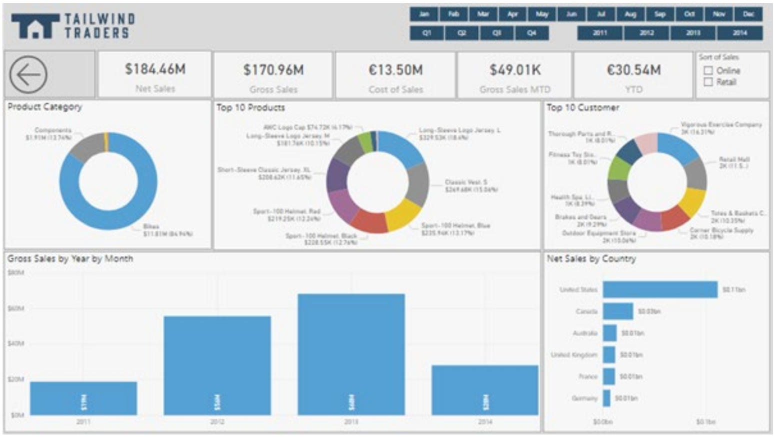


If you want to take a close look at the DAX query, select **Copy query** and then paste it into DAX Studio, for further analysis. DAX Studio is a free, open-source tool provided by a third party that you can download and install on your computer.

Tune performance

The results of your analysis will identify areas for improvement and highlight items that you need to optimize.

A common reason for poor performance is too many visuals on the same page. In the following image you can see an example of a busy page that contains many visuals.



If you identify visuals as the bottleneck leading to poor performance, you can take the following steps to tune the report:

- Reduce the number of visuals on the report page; fewer visuals means better performance. If a visual is not necessary and does not add value to the end user, should remove it. Rather than using multiple visuals on the page, consider other ways to provide additional details, such as drillthrough pages and report page tooltips.
- Reduce the number of fields in each visual. The upper limit for visuals is 100 fields, so a visual with more than 100 fields will be slow to load (and look cluttered and confusing). Identify fields that are not valuable to the visual and remove them.

If you find that visuals are not causing the performance issues, you should take a close look at the DAX Query results that are displayed in the **Performance analyzer** pane, and investigate further into those. For example, you might need to look elsewhere in your data model, such as the relationships and columns.

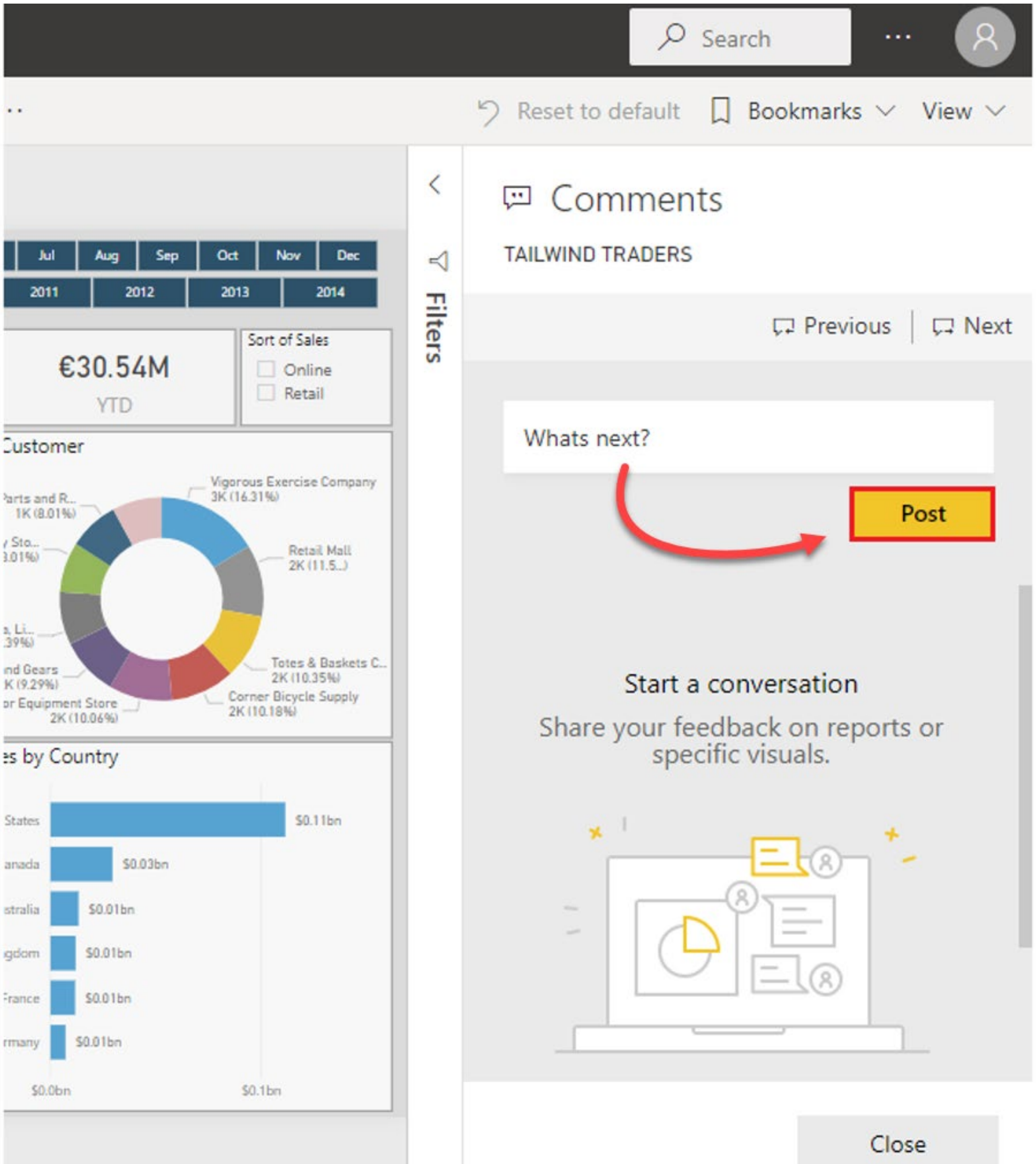
When you have made all your changes to performance tune the report, and you believe the report is performing well from your perspective but some users are experiencing poor performance, there might be other factors impacting on the performance. These factors include the bandwidth, server, firewall, and other external, uncontrollable factors. You might need to speak to the Information Technology (IT) Team in your organization, to see if they can shed any light on why users are experiencing poor performance when using your reports.

Commenting on reports

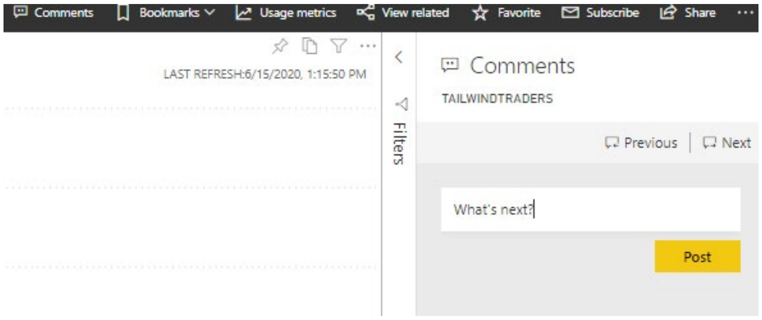
When you publish your report to Power BI web service, the consumers (users) of your report can add comments to the report. Comments can be useful for personal comments or for starting a conversation about a report item with your colleagues. For example, users can comment on pages or visuals that are experiencing issues with or they could give you suggestions for changes or improvements.

Comments are also available for paginated reports, dashboards and visuals. Anyone with the right permissions can see those comments. When you add a comment to a specific visual rather than the report as a whole, the context of the comment is more clear, and acts like a personal bookmark.

To add or view comment on a report, open the report in the Power BI web service. In the upper-right corner, select **Comments**. In the **Comments** pane, you can see any existing comments and write your own comments, then select **Post Comment**.



To add a comment from a visual, select the visual in the report to open that visual in focus mode. Then select **Comments** from the top menu, and type your comment in the **Comments** pane that displays.



Advanced interactions and drillthroughs

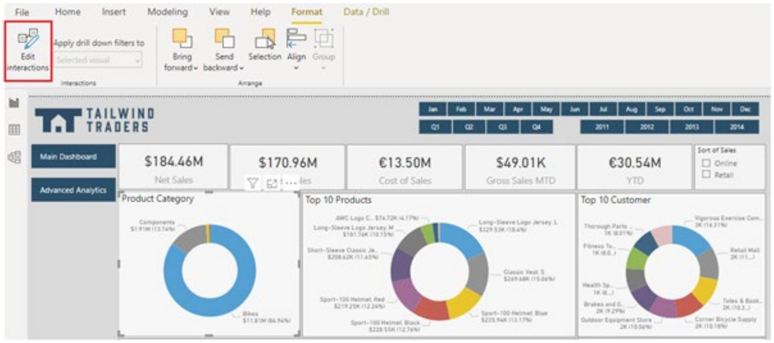
To ensure you have full control over the behavior of your report and can determine the expected user experience, you can edit the default interactions and use the drill-through features.

Edit interactions

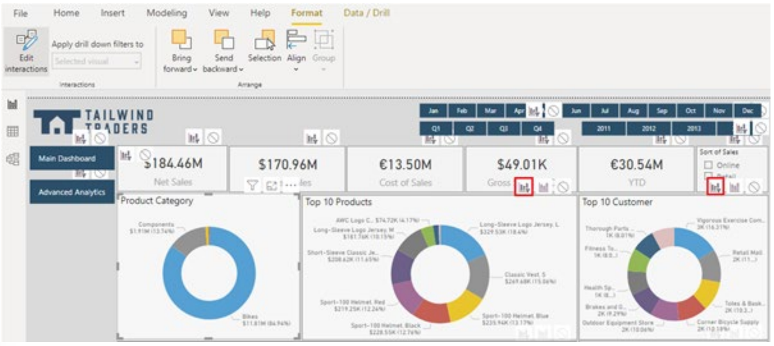
You can use visual interaction controls to customize how the visualizations on your report page impact each other, in order words, filter and highlight each other.

For example, when you select an item in a visual, the other visuals update to display data for that item. They might be highlighting or filtering the selected data. You can stop this from happening, or change a highlight action to a filter action and vice versa. Therefore, you have the power to determine what data is displayed for a specific selection that you've made.

To enable the visual interaction controls, select a visualization then go to the **Format** tab in the ribbon and select **Edit interactions**.

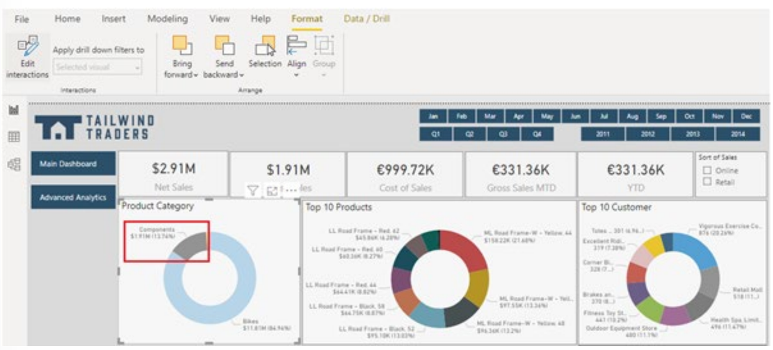


The **Edit interactions** button turns grey to show that is enabled, and **Filter**, **Highlight** and/or **None** icons are added to the other visualizations on the report page. When you hover over an icon, a grey box displays over the related visual. The bolded icon is the one that is being applied. In the following image, you can see that the tree map is cross-filtering the card visuals, and it is cross-highlighting the column chart. You can now change how the selected visualization interacts with the other visualizations on the report page.



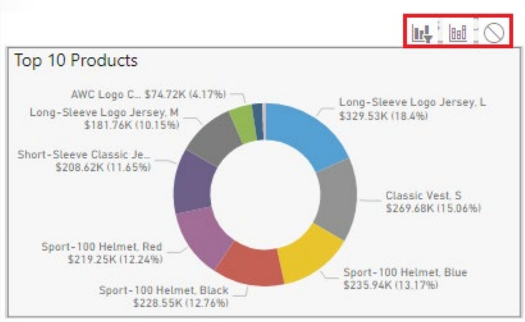
You can now select each visualization on your report page, one at a time, to see how it interacts with the other visualizations. If you do not like the behavior that you see, you can change the interactions. These changes are saved with the report, so your report users will have the same visual interaction experience as you do.

In the following example, you can see that all elements get updated once the Components category is selected. You can compare this image to the first image in this unit, when no category was selected.



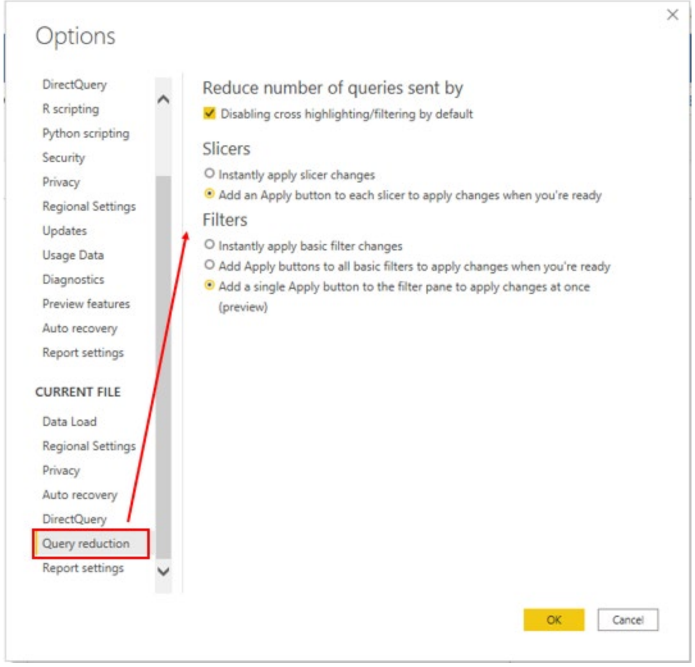
To change an interaction of a selected visual, select the required interaction icon. Remember, the applied interaction is bold. In the following image:

- The selected interaction is cross-highlight because the **Highlight** icon is bolded.
- You can change the interaction to cross-filter by selecting the **Filter** icon.
- You can remove the interaction altogether by selecting the **None** icon.



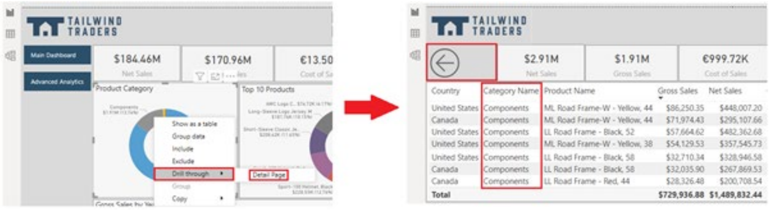
Keep in mind that the number of interactions between your visuals will impact on the performance of your report. To optimize the performance of your report, you should consider the query reduction options that are available within Power BI Desktop. You have the option to send fewer queries (which will reduce query chattiness) by disabling cross highlighting/filtering by default. You can also disable certain interactions that would result in a poor experience, if the resulting queries take a long time to run.

You access the query reduction settings by selecting **File** > **Options and settings** > **Options**, then scrolling down and selecting the **Query reduction** option.

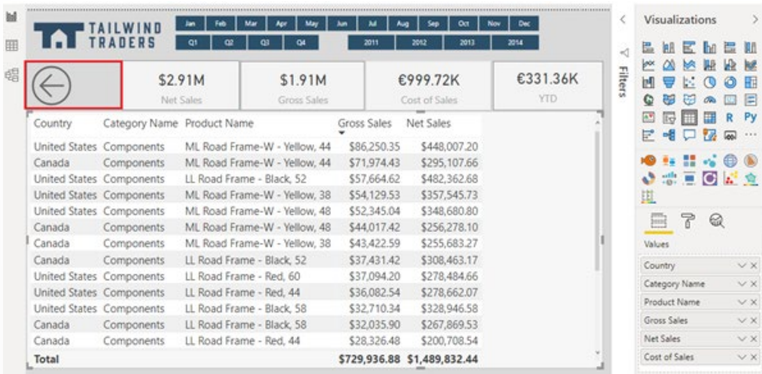


Drill through

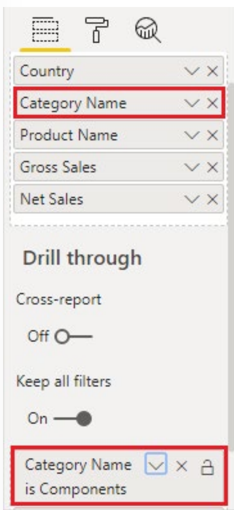
You can use the drill through feature to create a page in your report that focuses on a specific entity, such as a product, category, or region. You can then access this page when you drill through from the related visuals that are on other pages in your report. The information that displays on the drill through page will be specific to the item you select on the visual, as illustrated in the following image:



In this example, you create a drill through for the product category entity. You start by creating a page in your report and rename it to *Detail Page*. On that page, you add a visual for the entity that you want to provide the drill through for (a table that displays data for the **Category**, **SubCategory**, **Country** and **Gross Sales and Net Sales** fields).

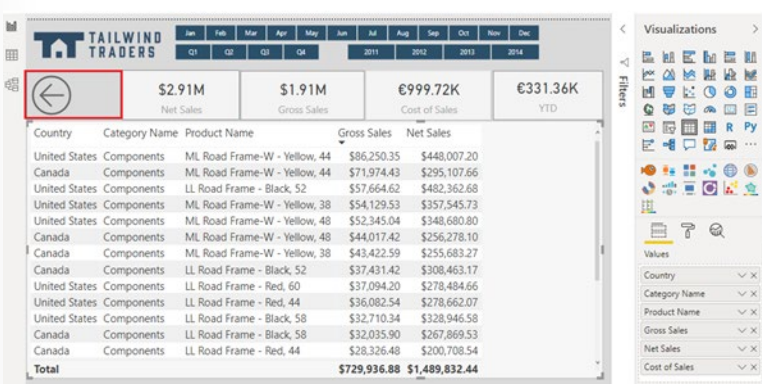


Then, from **Values** section of the **Visualizations** pane, drag the field (**Category Name**) for which you want to enable drill through into the **Drill-through filters** well.

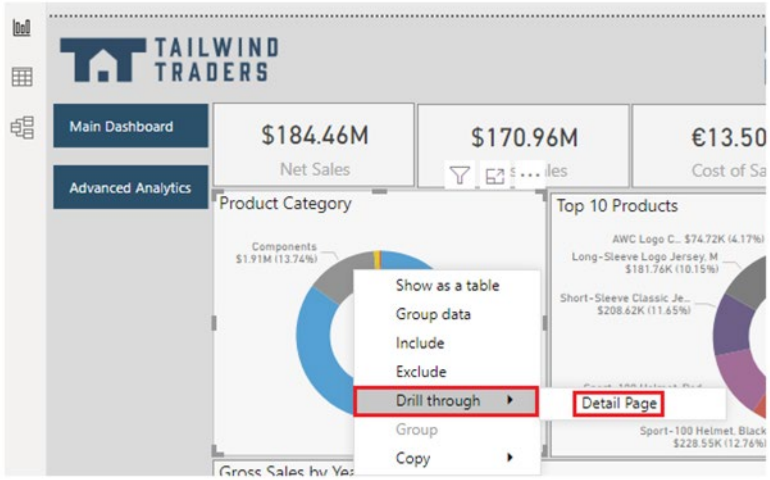


Ensure the **Keep all filters** option is set to **ON**, so when you drill through from a visual, the same filters will be applied on **Details** page.

Power BI Desktop automatically creates a **Back** button visual on the page for you. This button is for navigation purposes, so your report users can get back to the report page from which they came. You can reposition and resize this button on the report page or replace it with your own type of button.



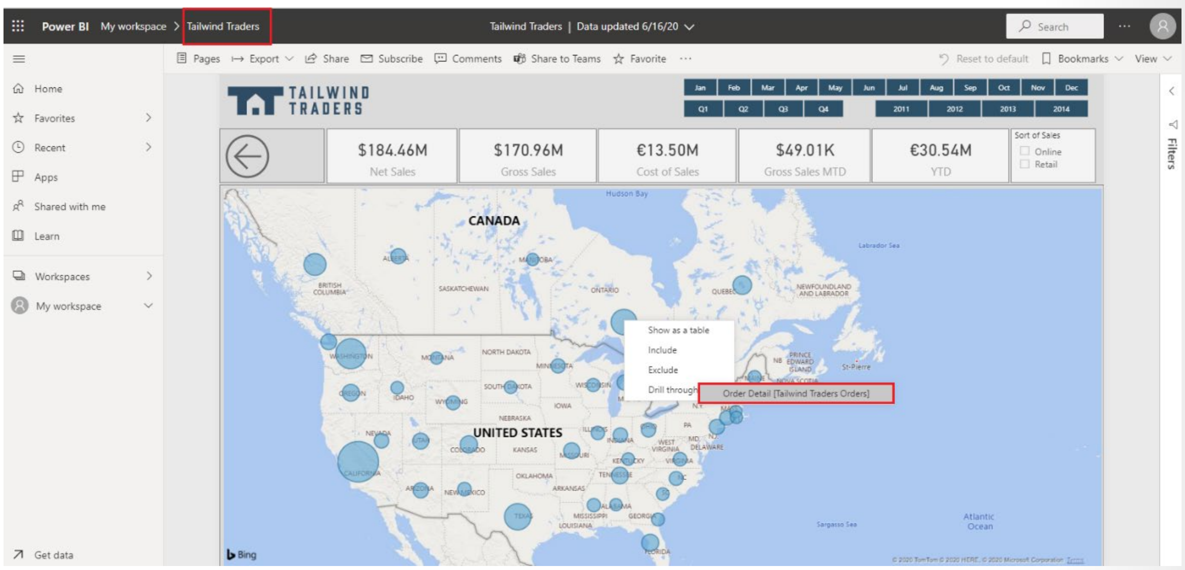
To use drill through, you right-click a data point on a visual in another report page, select **drill through**, then select the focused page (**Details page**) to get details that are filtered to that context.



Cross-report drill through

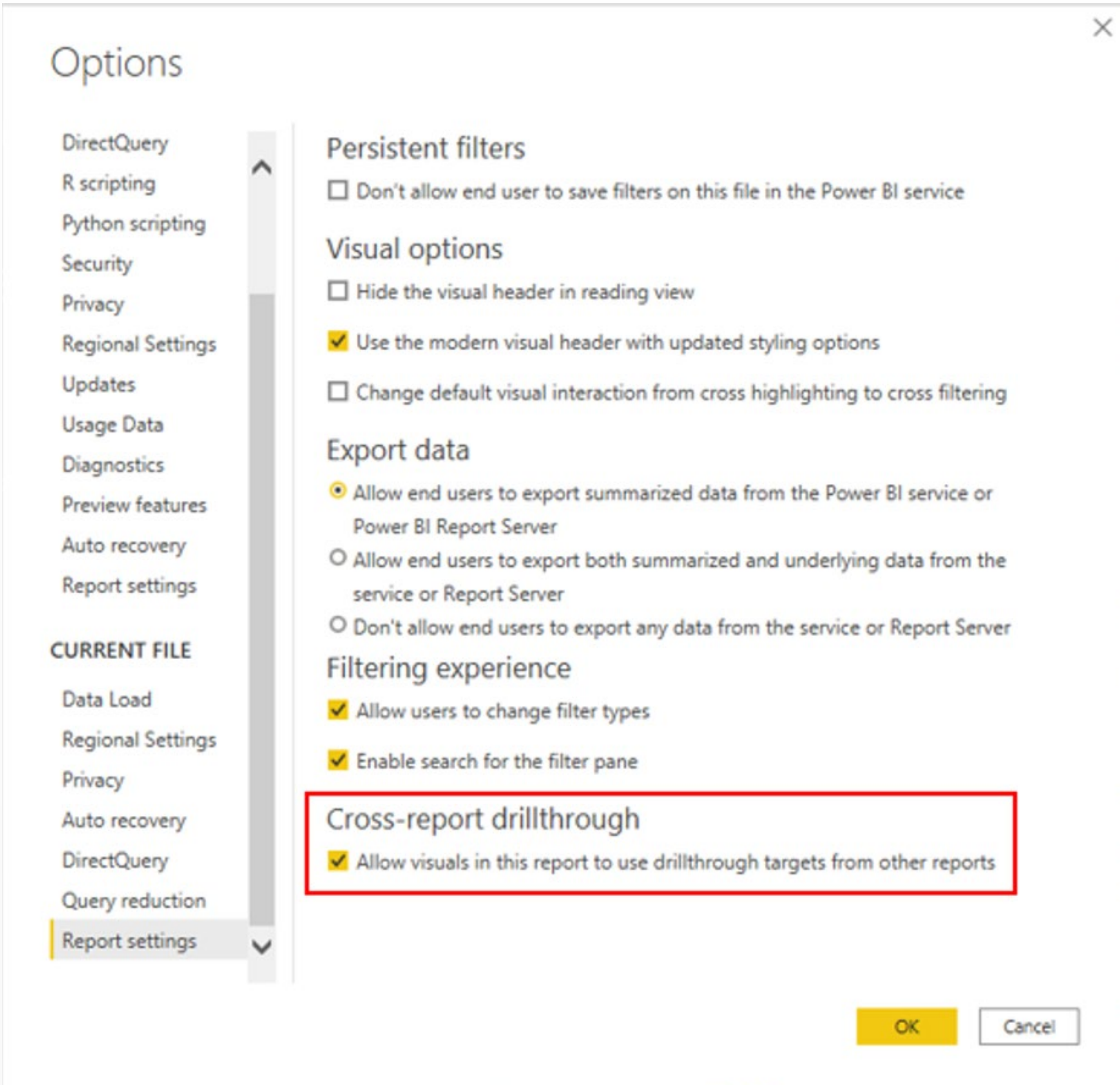
The **Cross-report drillthrough** feature allows you to contextually jump from one report to another report in the same Power BI service workspace or app, so you can connect two or more reports that have related content. You can also pass filter context along with that cross-report connection.

For example, you can select a data point on a visual in one report, then drillthrough to related, detailed information that is in another report.



To enable cross-report drillthrough, you first need to validate the data models for the source and target reports. Although the schemas in each report don't have to be the same, both data models must contain the fields you want to pass. Also, the names of those fields, and the names of the tables they belong to, must be identical. The strings must match, and are case-sensitive. If that is not the case, you must update the field name or table name in the underlying model.

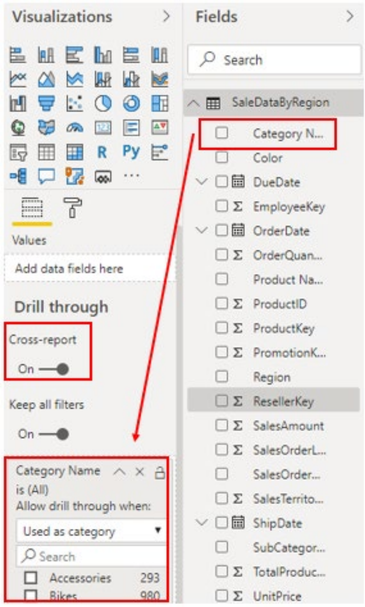
When you have validated your data models, you need to enable the **Cross-report drillthrough** feature in Power BI Desktop. Go to **File** > **Options and settings** > **Options**, then scroll down the **Current File** settings and select **Report settings**. In the **Cross-report drillthrough** section, select the check box for **Allow visuals in this report to use drillthrough targets from other reports** and select **OK**.



NOTE: The **Cross-report drillthrough** feature can also be enabled/disabled in the report settings in the Power BI Service.

Next, you set up a target page for the cross-report drillthrough, in the same way you set up a page for a drillthrough within a report in the previous section. The other visuals will target that page for drillthrough.

When you have set up the target page, go to the **Drillthrough** section of the **Visualizations** pane and set the **Cross-report** option to the **On** position. Then drag the fields you want to use as drillthrough targets into the **Drill-through filters** well. For each field, select whether you want to allow drillthrough when the field is used as a category, or when it's summarized like a measure.



Select whether you want the **Keep all filters** option **On** or **Off** for the visual. If you don't want to pass filters applied to the source visual to your target visual, select **Off**.

Just like when you create a drillthrough for a single report, Power BI Desktop automatically adds a **Back** button to the target drillthrough page. In this case though, you should delete the **Back** button because it only works for navigation within a report.

When you save and publish your changes, you can use cross-report drillthrough. Select the source report in the Power BI service, and then select a visual that uses the drillthrough field in the way you specified when you set up the target page. Right-click a data point on the visual, select **Drillthrough**, and then select the drillthrough target. You'll see that cross-report drillthrough targets are formatted as Page name [Report name].

Add buttons, bookmark, and selections

You can use the **Bookmarks**, **Buttons** and **Selections** features in Power BI Desktop to make your report more interesting and interactive, and easier for users to navigate.

- **Bookmarks** capture the currently configured view of a report page, so you can quickly return to that view later. You can use bookmarks for different reasons. For example, you can use them to keep track of your own progress when you are creating reports. You can also use them to build a PowerPoint-like presentation that goes through the bookmarks in order, thereby telling a story with your report.
- **Buttons** create a more interactive experience for the report users. With the addition of buttons that have assigned actions, your report behaves similar to an app, where users can hover, click, and interact more with the content.
- **Selections** allow you to determine what items in the report are visible and what items are hidden. Selections are used alongside bookmarks and buttons.

Suppose you have designed a report page and you want to have a second page that looks almost the same, except that one of the visuals is different. Rather than creating a second page and manually making changes, you can use a combination of selections, bookmarks and buttons to switch between the two visuals on the one page.

Add bookmarks

When you add a bookmark, the following elements are saved with the bookmark:

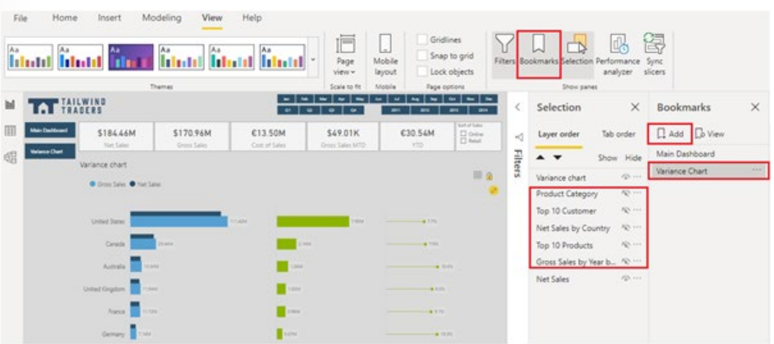
- Current page
- Filters
- Slicers, including slicer type (for example, dropdown or list) and slicer state
- Visual selection state (such as cross-highlight filters)
- Sort order
- Drill location
- Visibility of an object (by using the Selection pane)
- Focus or Spotlight modes of any visible object

In this example, you want to add bookmarks to allow users to switch between two visuals on one page. You first need to set up the page how you want it to display initially.

When you have added and formatted all of the visuals and other items on the page, you can add a bookmark to capture a snapshot of the page in its current state.

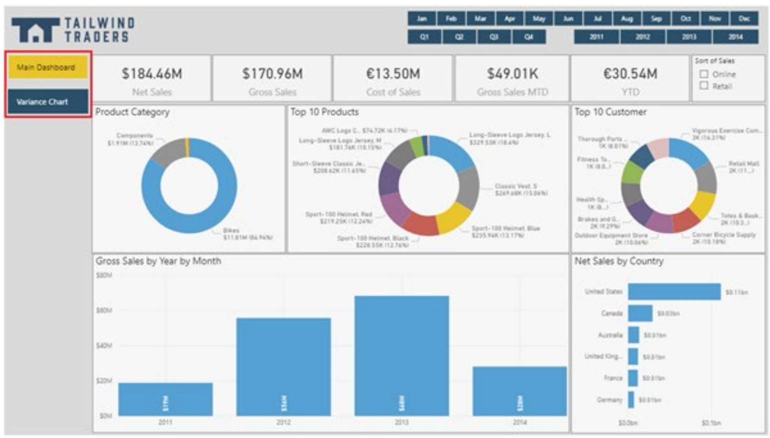
Before adding a bookmark, go to the **View** tab in the ribbon and select **Selection**. In the **Selection** pane that displays, you'll see a list of all of the items on your page, along with an eye icon that indicates the items that are currently visible. You can rename the items in the list by double-clicking on them, so you clearly know which one is which.

Now you are ready to add the bookmark. Again on the **View** tab, select **Bookmarks**. On the **Bookmarks** panel that displays, select **Add**. It's good practice to rename the new bookmark, so its purpose is clear, this is especially true if you plan on adding multiple bookmarks. To rename a bookmark, simply double-click the bookmark and enter the new name. In this example, you want to change the bookmark name from *Bookmark 1* to *Variance Chart* because the **Variance** chart is the main focus of the page, as you can see in the following image. In the **Selection** pane the **Variance** chart is displaying and the other visuals are hidden.



Now it's time to make the second bookmark. You start by making changes to how the page currently looks. In this example, you add another bookmark for the main dashboard charts. As you only want to see charts from the main dashboard, you hide the **Visual** chart by selecting its eye icon on the **Selections** pane. You then add a bookmark for this new view of the page, and rename it as **Main Dashboard**.

Now you can switch between the two bookmarks to see the difference in the page.

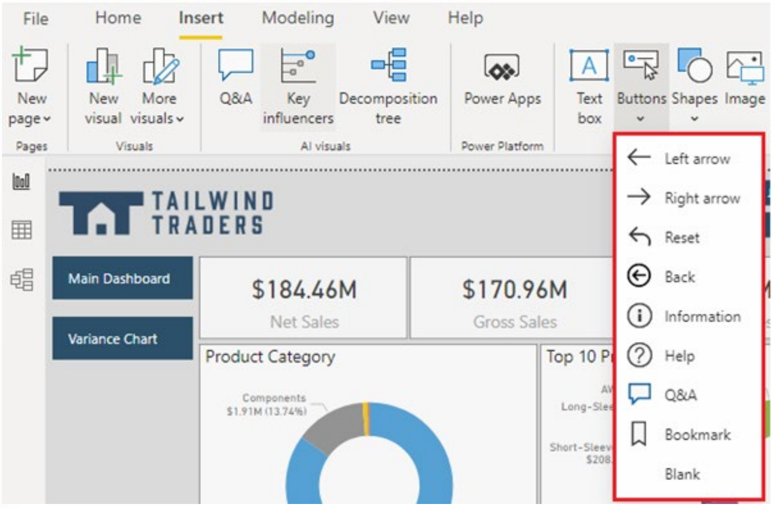


You can repeat these steps to add more bookmarks. In summary, you select items you want to show/hide on the report page, then add a bookmark and give it a descriptive name.

You can now assign those bookmarks to buttons to allow users to switch between the bookmarks. You'll learn how to add buttons in the next section.

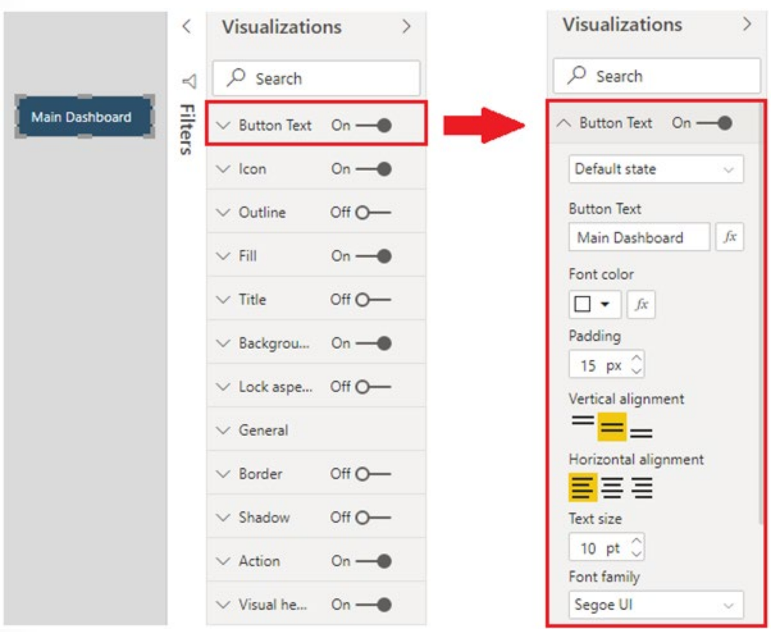
Add buttons

You can use buttons for many reasons, such as to switch between two visuals in a report (as required in the previous example), to drill-down into the data in a visual or to move from one page in your report to another. Power BI Desktop provides a range of types of buttons you can add to your report, as illustrated in the following image.



In this example, you want to add customized buttons that are used to switch between two bookmarks. To add a button, go to **Insert** tab on the ribbon and select **Buttons**, then select the type of button you want to add from the list, in this case you select the **Blank** option.

When the button is added to the page, reposition the button to above the visual, on the right side. Next, select the button, and then in the **Visualizations** pane, move the **Button Text** slider to the **On** position. Expand the **Button Text** section, and then type the text you want to display on the button, for example *Main Dashboard*. You can then format the button text by changing its font, color, alignment and text size. Expand the **Background** section and select a suitable color for the button.

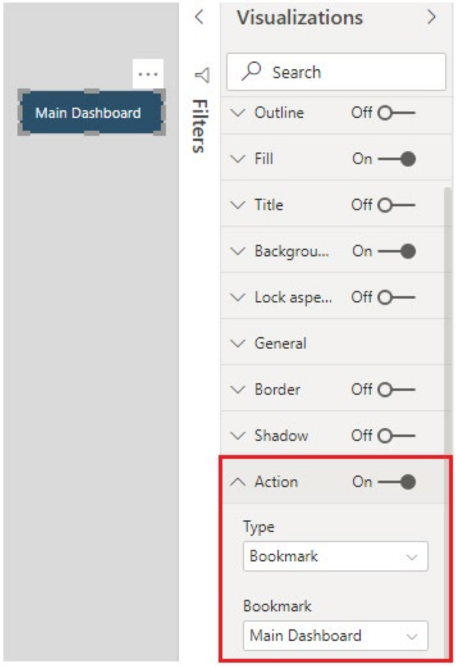


Now you add an action to the button. Go to the **Action** slider near the bottom of the **Visualizations** pane and move the slider to the **On** position, then expand the **Actions** section to view the options. The options for the button action types are listed below, and some of these options are explained in more detail in the subsequent sections.

- **Back** - Returns the user to the previous page of the report. This option is useful for drill-through pages or pages that are all accessed from one main page.

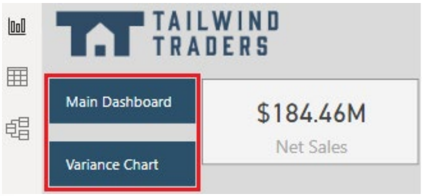
- **Bookmark** - Presents the report page that's associated with a bookmark that is defined for the current report.
- **Drill through** - Brings the user to a drill-through page that is filtered to their selection, without using bookmarks.
- **Page navigation** - Brings the user to a different page within the report, also without using bookmarks, which is an effective way to create a navigation experience for your report users. You will take a closer look at this type of button in the next unit.
- **Q&A** - Opens a **Q&A Explorer** window, which allows users to type questions to quickly find the information they are looking for, and specify the type of visual they want to see that information displayed in. This option can be useful if you want to save space in the report but still offer **Q&A** functionality to the user.
- **Web URL** - Opens a website in a new browser window. For example, you might want to give users quick access to your organization's website or Intranet from within a report.

In this example, you select **Bookmark** as the action type. In the **Bookmark** list that displays, select the bookmark that you want to assign to the action - the **Main Dashboard** bookmark you created earlier. For enhanced accessibility, you should add a tooltip that users will see when they hover over the button but in this case, it's not required, as you have a descriptive button label.



Now that your button is all set up, you can copy and paste the button, so you have two buttons with consistent formatting on the page. In this example, you rename the second button to *Variance Chart* and change the assigned action to the **Variance Chart** bookmark.

You can then reposition the buttons anywhere on your canvas but in this example, you position the new buttons on the left side of the canvas, for easy navigation.



Your buttons are now ready for use. You can use them to switch between two bookmarks with a different layout on the same page. When you select the **Main Dashboard** button, the main dashboard charts display, and when you select the **Variance Chart** button, the **Variance** chart displays.

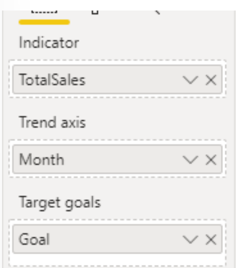
Key Performance Indicators

Key performance indicators (KPIs) are excellent in helping you track progress toward a specific goal over time. To use a KPI, you need three pieces of information:

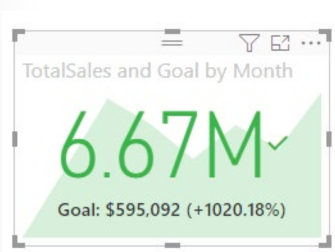
- A unit of measurement that you want to track, for instance total sales, number of employee hires, number of loans serviced, or number of students enrolled.
- A goal for the measurement so that you can compare your progress with that goal.
- A time series, for instance daily, monthly, or yearly.

Start by adding the KPI visual to the design service. The following screenshot shows the KPI icon in the **Visualizations** pane.

When configuring the KPI visual, enter the unit of measurement that you are tracking in the **Indicator** prompt. Then, enter the goal under **Target goals** and select the time series from the **Trend axis** drop-down list, as shown in the following screenshot.



This action will produce a KPI that looks similar to the following screenshot.



KPIs work best in a series, for instance, showing the daily, monthly, and yearly goals in the section of a Power BI report.

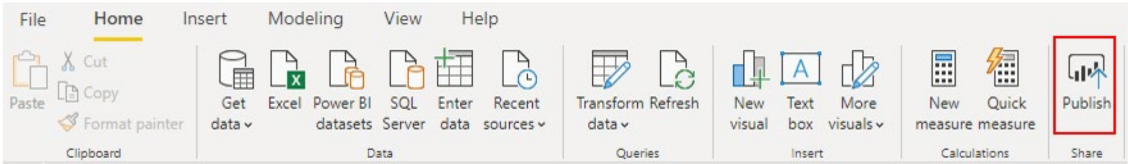
Publish and export reports

When you are finished designing your report, you can publish the report to your Power BI workspace. There are also options to export to Microsoft Excel.

Publish reports

When you publish a report, Power BI Desktop packages your report and data, including all your visualizations, queries, and custom measures, and uploads them to the Power BI service.

To publish your report, select the **Publish** button on the **Home** tab.



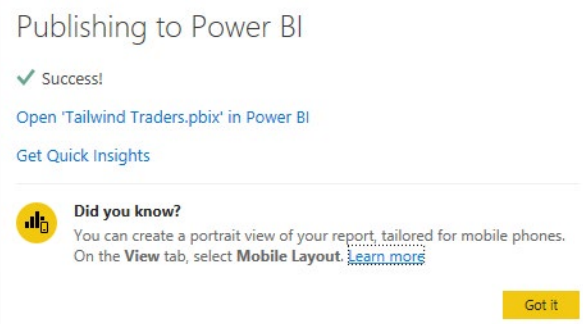
You might be prompted to save your changes, in which case, select **Save**, and proceed to save your Power BI (.pbix) file.

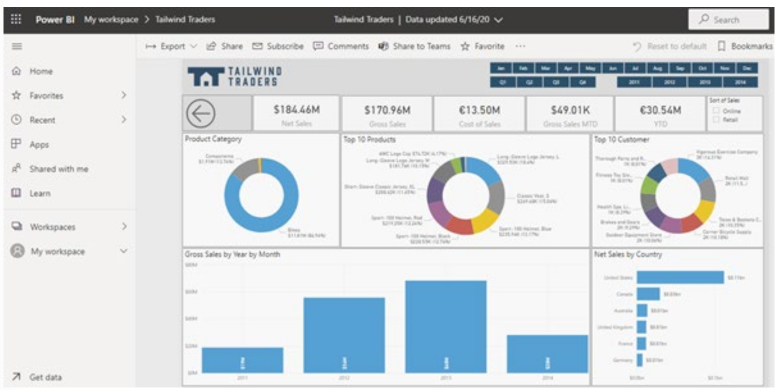
You might also be required to sign in to Power BI, in which case, enter your sign in credentials to continue.

On the **Publish to Power BI** window, select the destination in which you want to publish the report. For example, you can publish to a workspace within Power BI. For production reports, it's recommended to publish to an App Workspace.

When the report is successfully published, you'll get a success message that contains a link to your report in your Power BI site. Select **Got it** to close the **Publishing to BI** window and return to your report in

Power BI Desktop. From here, you can click on the URL provided to navigate to the Power BI Service and see your new report.





Export reports

Power BI allows you to export visual data, reports, and datasets. You can export to several different formats including CSV, Excel, and PDF.

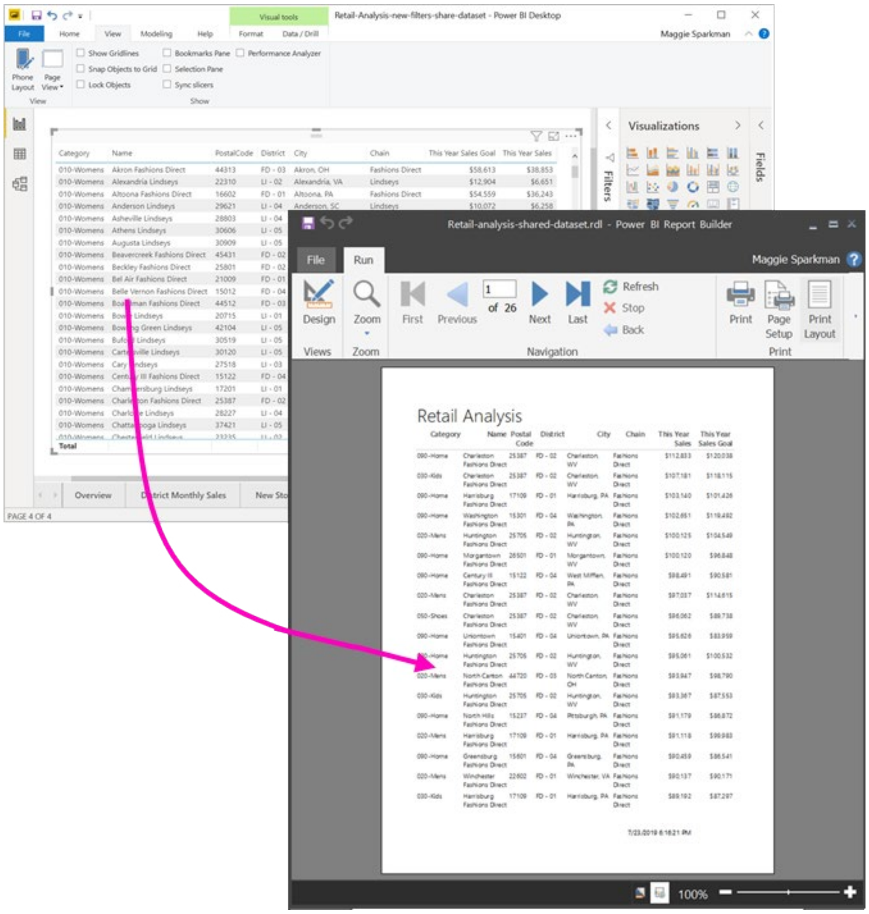
Introduction to paginated reports

Paginated reports allow report developers to create Power BI artifacts that have tightly controlled rendering requirements. Paginated reports are ideal for creating sales invoices, receipts, purchase orders, and tabular data. This module will teach you how to create reports, add parameters, and work with tables and charts in paginated reports.

Paginated reports defined

Paginated reports give a pixel-perfect view of the data. Pixel perfect means that you have total control of how the report renders. If you want a footer on every sales receipt that you create, a paginated report is the appropriate solution. If you want a certain customer's name to always appear in green font on a report, you can do that in a paginated report.

Power BI paginated reports are descendants of SQL Server Reporting Services (SSRS), which was first introduced in 2004. Power BI paginated reports and SSRS have a lot in common. If you're are looking for information on paginated reports and can't find it, searching the internet and Microsoft documentation on SSRS is an excellent idea because you'll find numerous blog posts, videos, and documentation available to you.



In this module, you will:

- Get data.
- Create a paginated report.
- Work with charts and tables on the report.
- Publish the report.

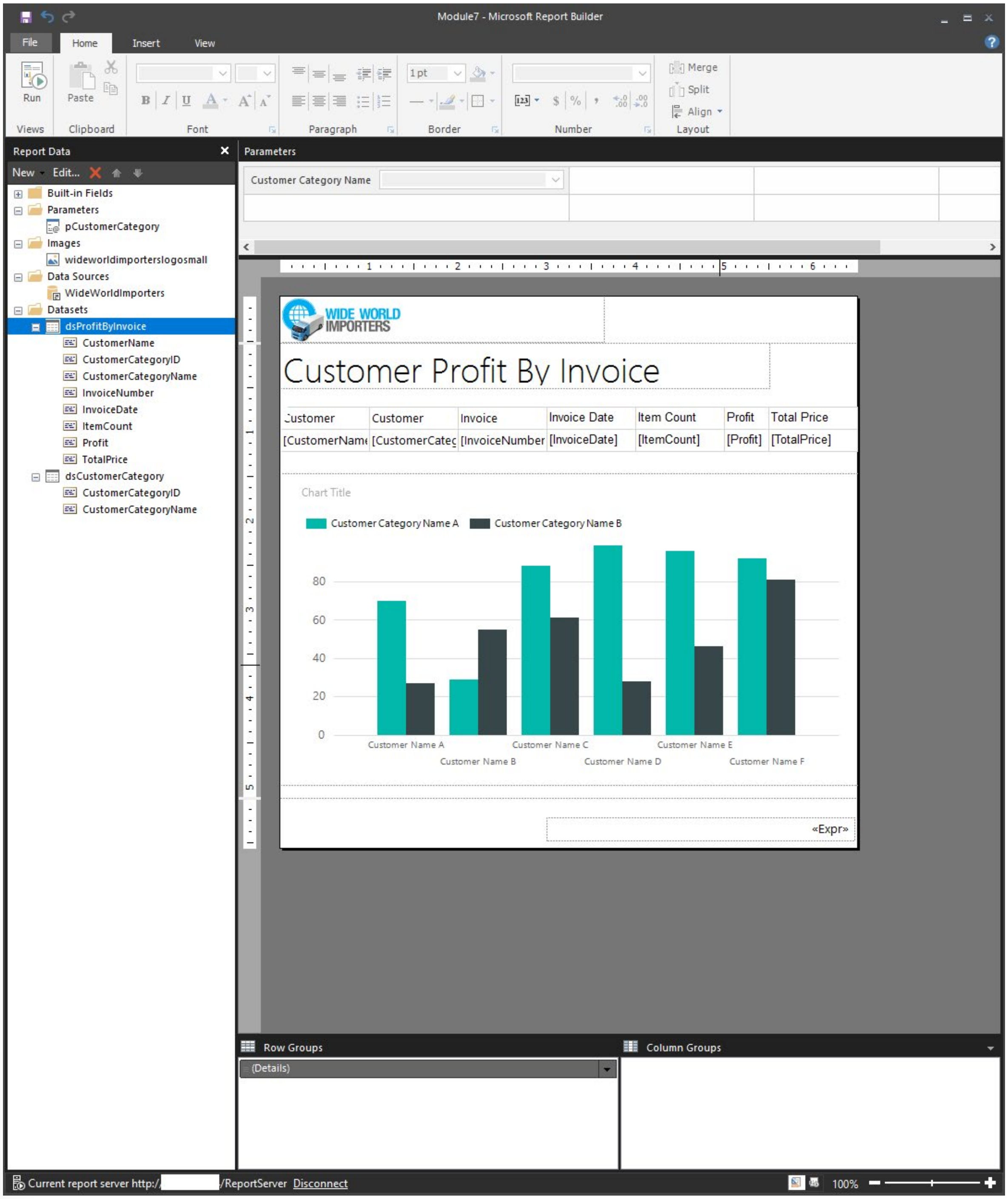
When are they the right fit

You can use paginated reports for operational reports with tables of details and optional headers and footers.

Additionally, you can use paginated reports when you expect to print the report on paper or when you want an e-receipt, a purchase order, or an invoice. Paginated reports also render tabular data exceedingly well. You can have customized sort orders, clickable-headers, and URLs in results, which allows for simple integration with custom applications.

Power BI paginated reports can also display all of your data in a single report element, such as a table. If you have 25,000 records, and you want the reports to print over 100 pages, you can do that. If you want every third record to be printed with a light pink background, you can do that as well.

Power BI paginated reports are not created in Power BI Desktop; they are built by using Power BI Report Builder. Power BI paginated reports are a feature of Power BI Premium.



Lesson Review

The managers in your organization were finding it difficult to obtain the information they needed to carry out their yearly planning and forecasting activities, and make good strategic decisions.

You were tasked with creating a concise, well-designed report that displayed information in an interesting way that was easy to navigate and more accessible. You needed to enhance the range of visuals on your report to offer more end-user interaction and detailed information.

Power BI Desktop report editor gave you all the tools you needed to create such a report. You started by considering the best position and size of your visuals and designed the report layout with user accessibility in mind. Next, you applied a selection of buttons, bookmarks, filters, and other elements to make the report more visually pleasing, interactive, and easier to navigate. You then took time to consider the interactions of the visuals in your report, and made some changes. You added a variety of slicing, filtering, and sorting techniques to your data, so users could more easily find the information they needed, at both a high and low detail level. At that stage, you were ready to publish the report to the Power BI Report Server, so it could be accessed by all end users. You then checked the performance of your report and made some changes to fine-tune the report for optimal performance. Lastly, you designed and published a mobile version of your report.

Imagine if you could not use Power BI Desktop to produce reports. You would be able to analyze the data but you would not be able communicate your findings. Power BI Desktop does more than enable you to communicate the data. It allows you to design a range of interesting, powerful reports that can be used for telling data drive stories and aid decision making at all organization levels.

Now that you have published your report, your managers have easy access to up-to-date data that can help them to make more robust plans and accurate forecasts, and ultimately make better business decisions.

Knowledge Check

Question 1

Which of the following filters are not available in Power BI reports?

- ☐ Drillthrough
- ☐ Report level
- ☐ Page type
- ☐ Page level

Question 2

How can you analyze performance of each of your report elements?

- ☐ By using performance analyzer.
- ☐ By analyzing your metadata.
- ☐ By deleting unnecessary rows and columns to reduce your dataset size.

Question 3

Can you use bookmarks to create a slide show in Power BI?

- ☐ No, you cannot, because bookmarks are not dynamic.
- ☐ Yes, you can.
- ☐ No, you will require a specific visual to achieve this task.

Answers

Question 1

What is the benefit of using a report tooltip?

- ☐ To give users the ability to export data from the visual.
- ■ To provide additional detail that is specific to the context of the data that is being hovered over.
- ☐ To give users additional information about a report visual, such as the author and date/time it was created.

Question 2

Do you need to import custom visuals each time you want to use them when you are developing a new report?

- ■ Yes, custom visuals must be imported from AppSource each time you start developing a new report unless you pin the custom visual to the visualizations pane.
- ☐ No, custom visuals are always available for selection under the Visualization pane.
- ☐ No, custom visuals only need to be imported once and will always remain in Power BI for future use in a new report.

Question 1

Which of the following filters are not available in Power BI reports?

- ☐ Drillthrough
- ☐ Report level
- ■ Page type
- ☐ Page level

Question 2

How can you analyze performance of each of your report elements?

- ■ By using performance analyzer.
- ☐ By analyzing your metadata.
- ☐ By deleting unnecessary rows and columns to reduce your dataset size.

Question 3

Can you use bookmarks to create a slide show in Power BI?

- ☐ No, you cannot, because bookmarks are not dynamic.
- ■ Yes, you can.
- ☐ No, you will require a specific visual to achieve this task.

Module 8 Create Dashboards

Create a Dashboard

Introduction to Dashboards

Microsoft Power BI dashboards are different than Power BI reports. Dashboards allow report consumers to create a single artifact of directed data that is personalized just for them. Dashboards can be comprised of pinned visuals that are taken from different reports. Where a Power BI report uses data from a single dataset, a Power BI dashboard can contain visuals from different datasets.

Well-built dashboards capture the main, most important highlights of the story that you are trying to tell. The following screenshot is an example of a well-built dashboard.

Power BI dashboards is a feature that is only included in Power BI service. You can also view dashboards on mobile devices, though you can't build them there.

Consider dashboards as the display window at a bakery, where you want people to be able to view the most important items, while inside the shop (and in your reports in Power BI Desktop) is where all ingredients are transformed to produce the display.

Dashboards vs. reports

When would you want to build a dashboard versus a report? The following list explains the key similarities and differences worth noting when you are determining the right path for you:

- Dashboards can be created from multiple datasets or reports.
- Dashboards do not have the **Filter**, **Visualization**, and **Fields** panes that are in Power BI Desktop, meaning that you can't add new filters and slicers, and you can't make edits.
- Dashboards can only be a single page, whereas reports can be multiple pages.
- You can't see the underlying dataset directly in a dashboard, while you can see the dataset in a report under the **Data** tab in Power BI Desktop.
- Both dashboards and reports can be refreshed to show the latest data.

Dashboards allow a user to pin visuals from different reports and datasets onto a single canvas, making it simple to group what's important to the user. Reports, on the other hand, are more focused on being able to visualize and apply transformations to a single dataset. Consider dashboards as the next step that you want to take after building your reports in Power BI Desktop.

Now that you've learned about the background of dashboards and reports, you can learn about dashboards in depth, specifically about their individual components.

Manage tiles on a dashboard

Tiles are the individual report elements, or snapshots, of your data that are then pinned to a dashboard. Tiles can be sourced from a multitude of places including reports, datasets, other dashboards, Microsoft Excel, SQL Server Reporting Services, and more. When pinning a report element to a dashboard, you create a direct connection between the dashboard and the report that the snapshot came from.

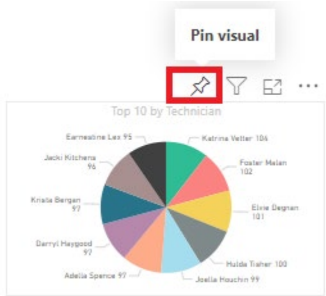
Your first task in this module is to create a basic dashboard. For this scenario, you have created a simple report in Power BI Desktop called **Tailwind Sales**.

Pin a tile to a dashboard

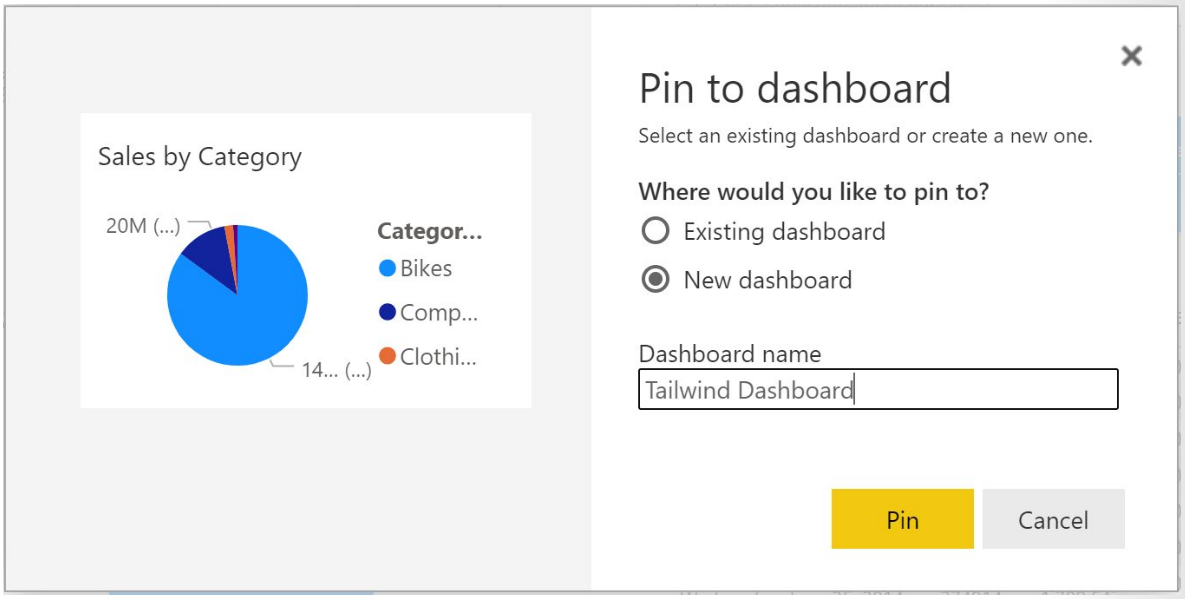
You've uploaded your reports into Power BI service and are now viewing the report in Power BI service. How do you create a dashboard? You can pin an entire report page, or you can pin individual tiles, both of which will be discussed later.

The pinning process pulls visuals from your report and “pins” them to a dashboard for easy viewing. When you make changes to any of the visuals in the report, changes will be reflected on the dashboard.

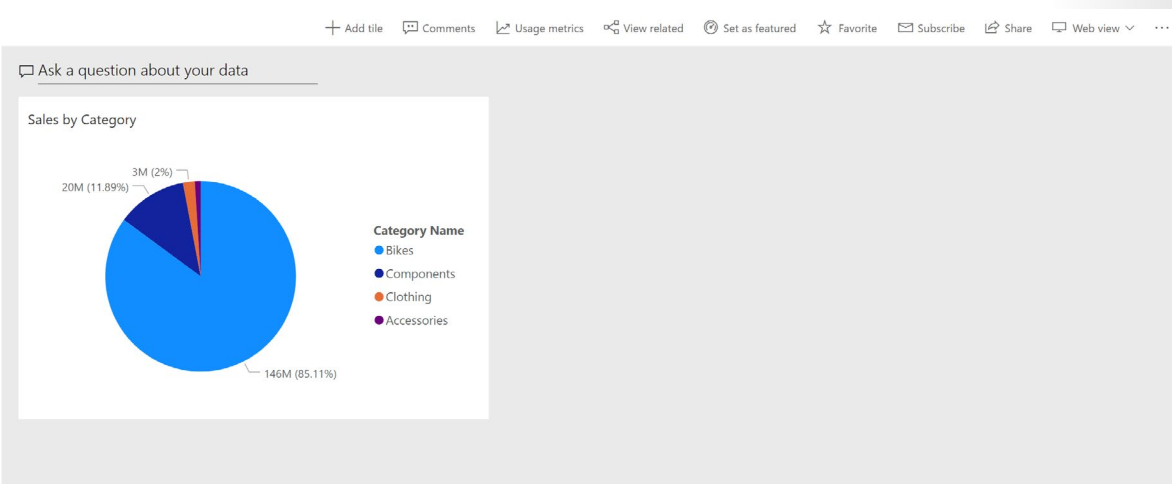
To look at a specific visual, consider that you want to pin your tile, **Sales by Category**, onto a new dashboard for easy viewing. You can complete this task by hovering over the visual. In the visual header, select the **Pin Visual** icon, as shown in the following image.



After you have selected the icon, a window will appear, where you can choose to pin this visual to a new or existing dashboard. For this example, you want your tile to be on a new dashboard called **Tailwind Dashboard**.



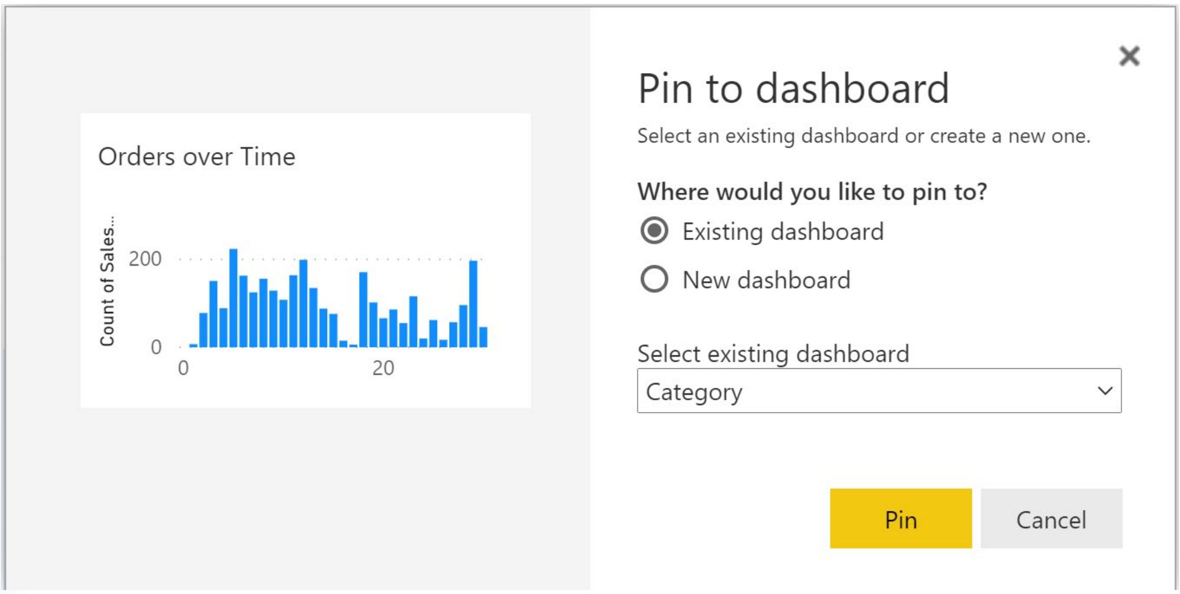
After you have selected **Pin**, you will be redirected to your new dashboard, where you have just pinned a tile from your report. You can resize and move this visual around the dashboard by selecting the visual, dragging, and then dropping it.



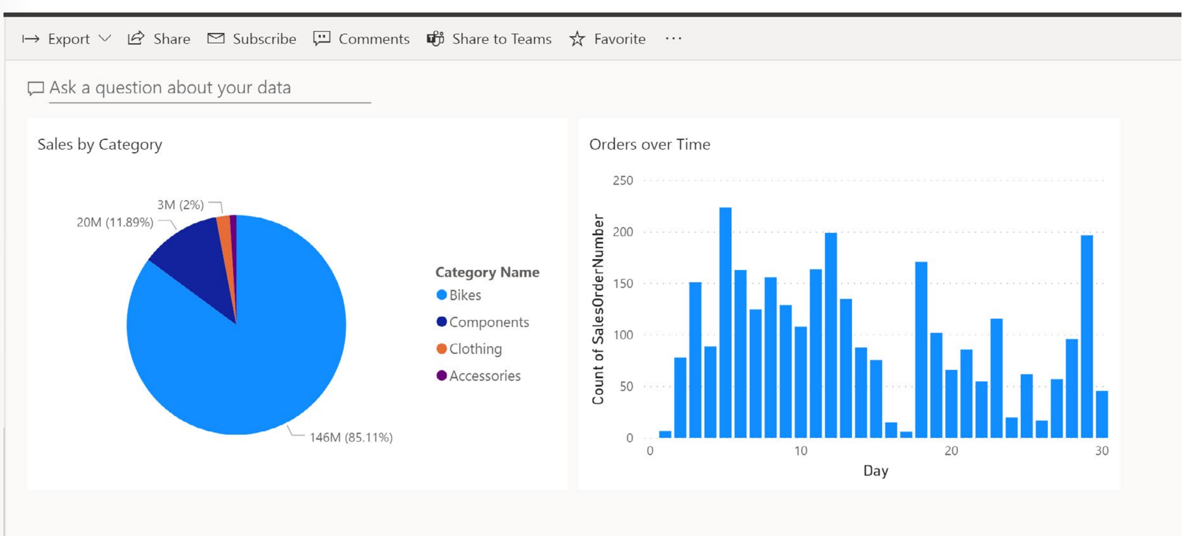
One of biggest benefits of a dashboard is being able to pin a visual that is sourced from a different dataset. The following section explains how you can add a visual onto your **Category** dashboard.

Pin a tile from a different report

What if you want to pin a visual from a different report (and different dataset) to an existing dashboard? To continue with the scenario, you want to add an **Orders over Time** visual, which is housed in a different report to **Tailwind Dashboard**. You can perform the same procedure in which you hover over the visual in the original report and then select the **Pin** icon. The following window will appear, but this time, you want to pin this visual onto an existing dashboard.



When you navigate to your dashboard, notice that both visuals are now pinned, regardless of the underlying dataset.



Now that you have learned how to pin individual tiles, you can learn how to pin an entire report page, which will be discussed later in this module.

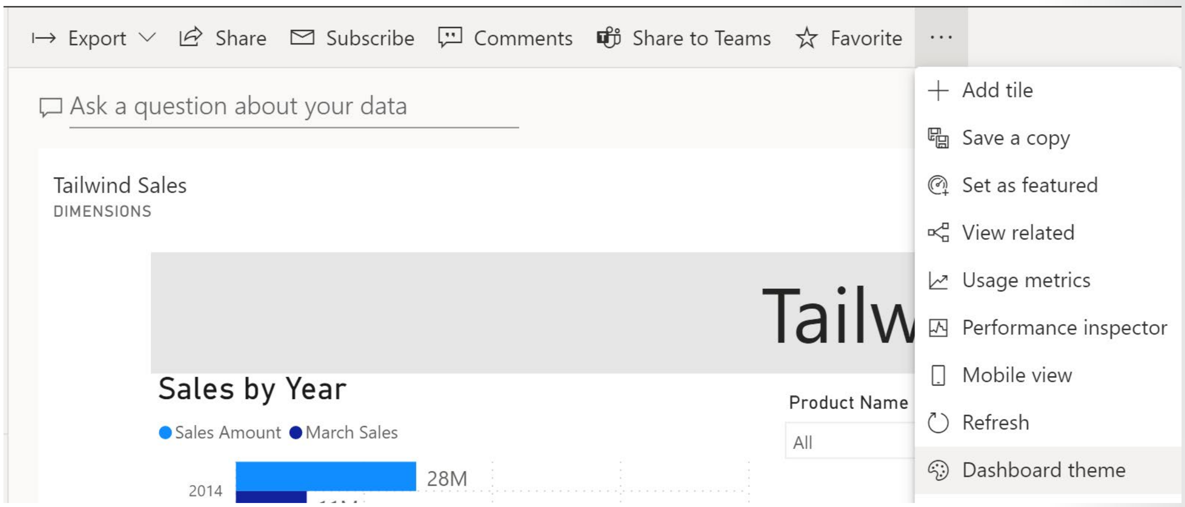
For more information, see **Introduction to dashboard tiles.**[1]

Add a dashboard theme

When building dashboards, you should consider ensuring that the same theme is applied to your dashboards to create a cohesive picture. You could also apply a specific theme to reports and dashboards so that all report elements or tiles are uniform. This consideration is particularly important when you are building multiple dashboards. Power BI has the functionality to apply a theme directly to all visuals of a report.

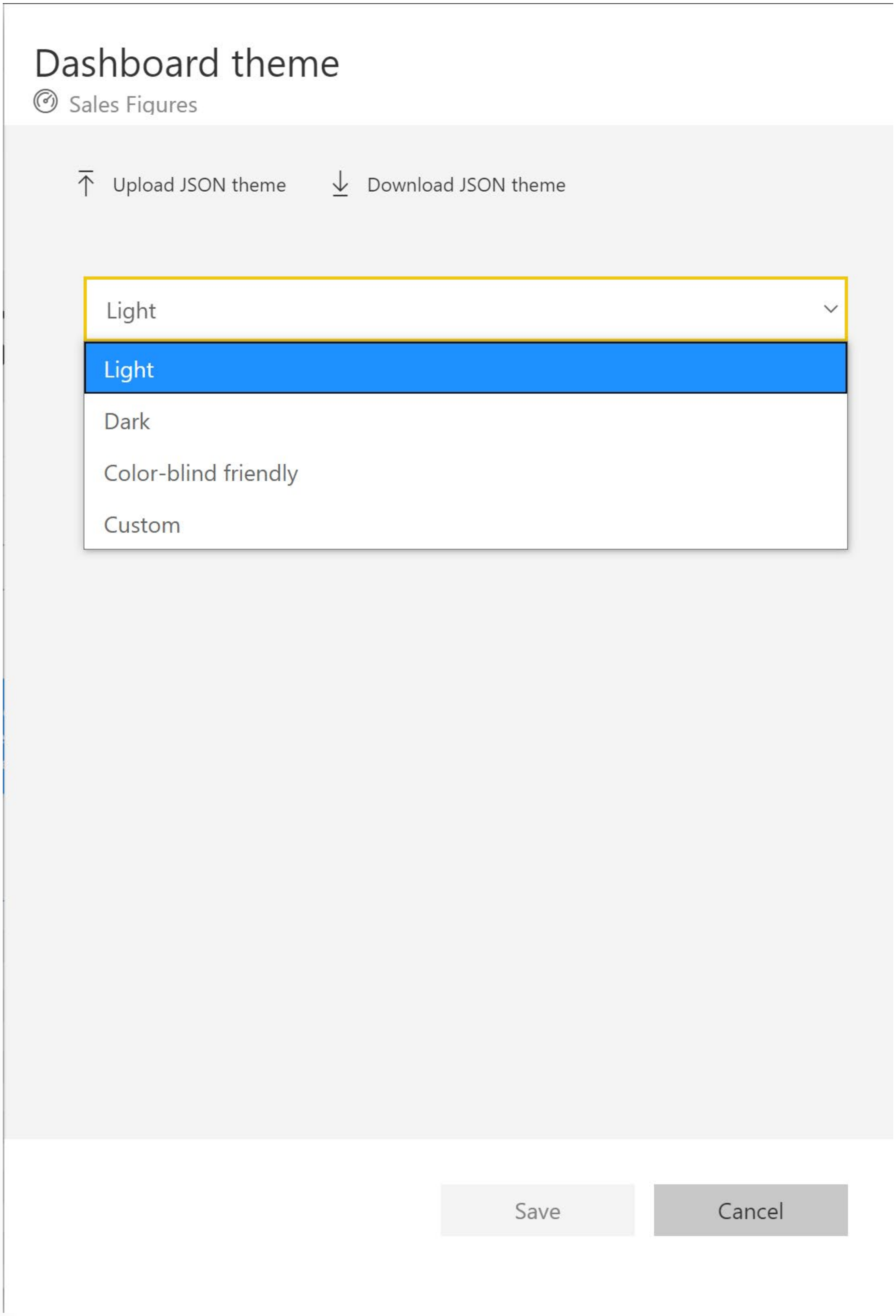
Themes in Power BI

A variety of themes are available for use in Power BI service. Go to a dashboard, select the ellipsis (**...**), and then select **Dashboard theme**.

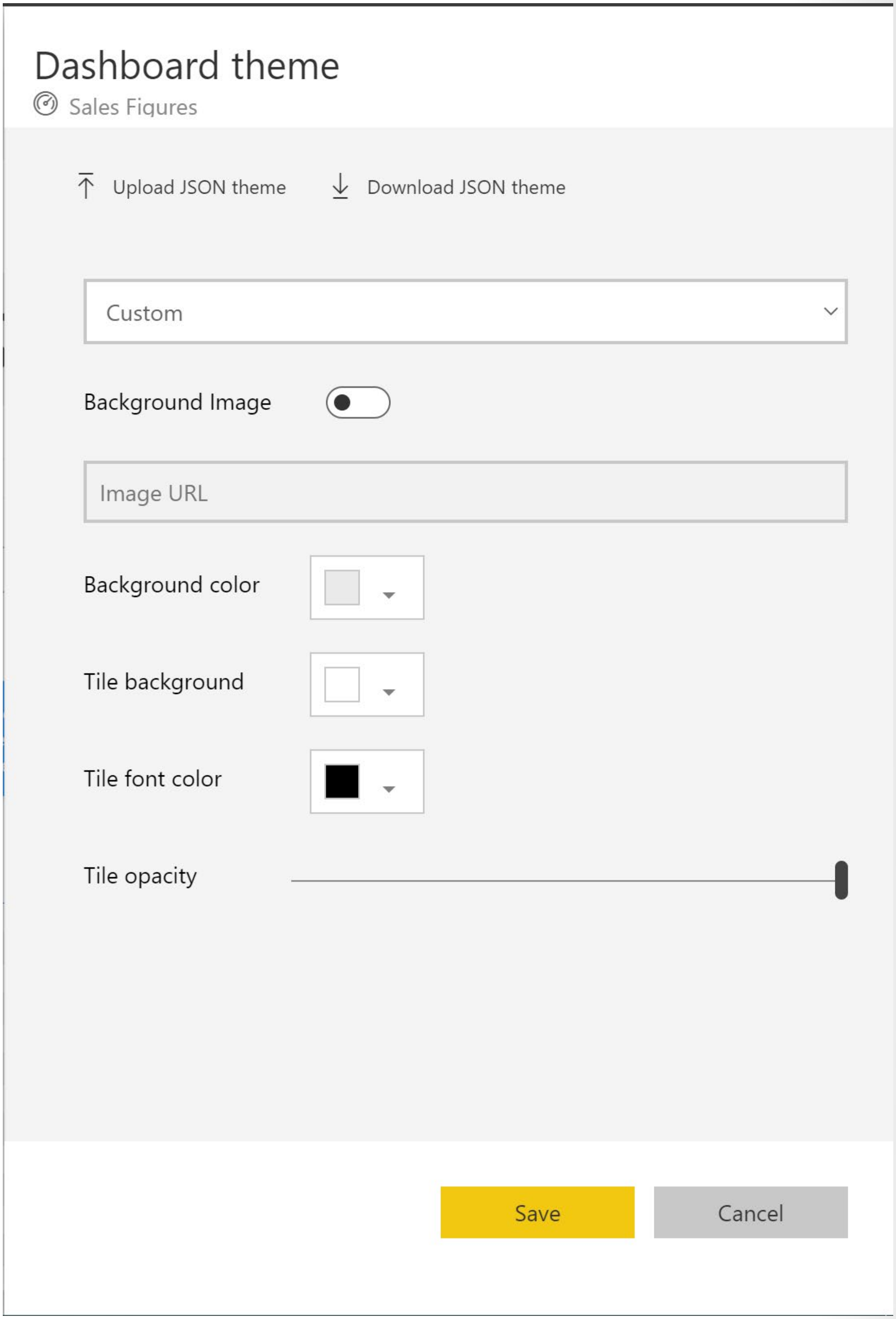


This selection will open a window, where you can choose from a variety of themes, including **Light** (the default theme), **Dark**, **Color-blind friendly**, and **Custom**, where you can create your own theme. You can also upload your own JSON theme or download the current theme.

1 https://docs.microsoft.com/power-bi/create-reports/service-dashboard-tiles


Dashboard theme
Sales Figures
Upload JSON theme
Download JSON theme
Light
Light
Dark
Color-blind friendly
Custom
Save
Cancel

For instance, if you select **Custom**, you can add your own background image, or you can change the background color, tile color, the opacity, or even the font color, as shown in the following figure.

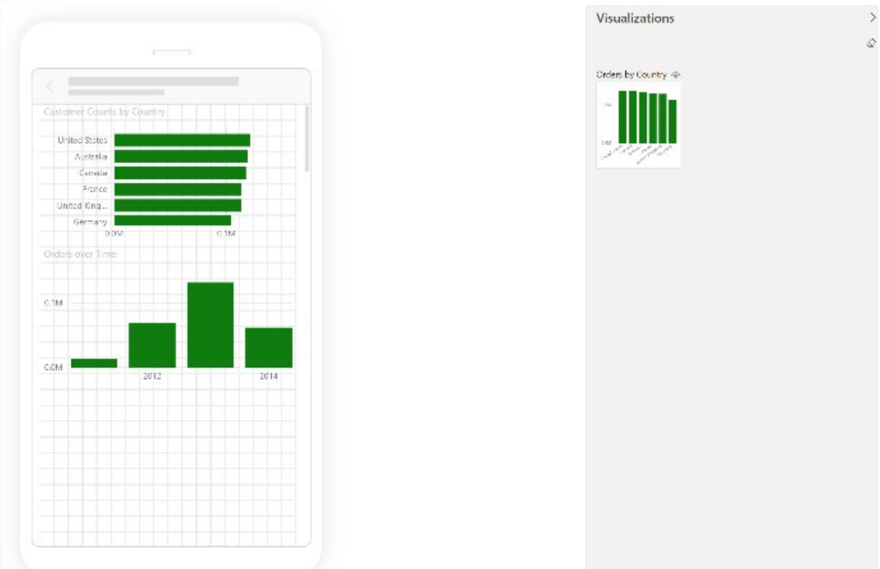


Now, you can customize your report to cater specifically to your needs.

Set mobile view

Power BI reports are built in Power BI Desktop and then deployed to Power BI service, where they can be viewed and shared. However, if you are building dashboards for the Sales team at your organization and you receive a requirement that the dashboards should also be viewable on mobile devices, Power BI will help you to set dashboards to mobile view.

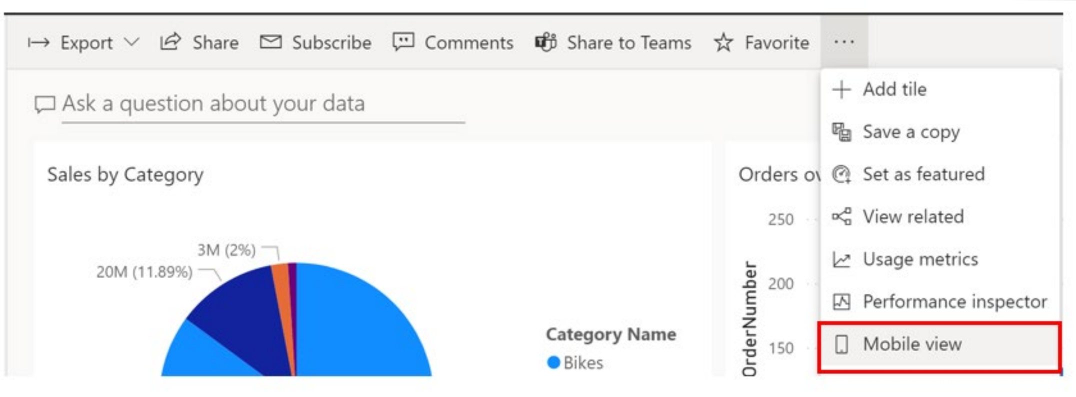
To navigate to mobile view in Power BI Desktop, select **View** on the ribbon and then select **Mobile Layout**, which will redirect you to the mobile view, as shown in the following figure.



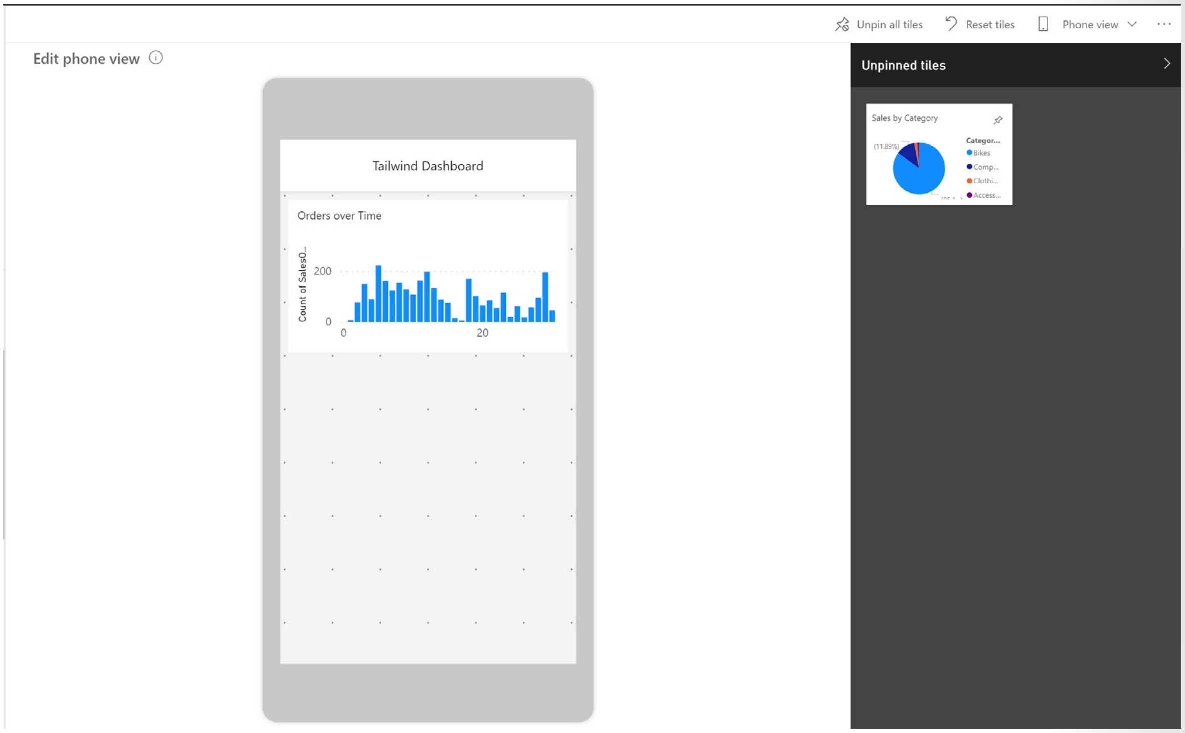
In the mobile view in Power BI Desktop, you are able to accomplish several tasks. This view emulates the view of a user who is looking at visuals on their phone, so you can add visuals to this view, resize them, and change the formatting on them, as shown in the ensuing screenshot. In the June 2020 release of Power BI Desktop, a new grid has been added to this view so that you can position your visuals with more ease and overlay visuals on top of each other. This feature can be useful if you want to insert a visual on top of an image.

After you have published to Power BI service, you can view your visuals on a mobile device.

Alternatively, you can also optimize your dashboards for mobile view in Power BI service. To see a dashboard in mobile view, select the ellipsis (**...**) on the home ribbon and select **Mobile view**, as shown in the following Sales dashboard.



This selection will take you to the following view, where you can choose which tiles that you want to see on the phone view.



You can also resize and reposition the tiles and visuals in whichever order you want. This phone view is customizable for each person who uses the dashboard; selecting **Phone view** will allow you to create a new view that you can see on your phone when signing in to Power BI service.

For more information, see **Optimize a dashboard for mobile phones**[2].

2 https://docs.microsoft.com/power-bi/create-reports/service-create-dashboard-mobile-phone-view

Knowledge Check

Question 1

What is a dashboard?

- ☐ A canvas of report elements that can be built in Power BI desktop.
- ☐ Dashboards can be built by using visuals that are developed with an underlying data source.
- ☐ A canvas of report elements that can be built in the Power BI service.
- ☐ The canvas in which you can view the data model of a report.

Question 2

What is one way that reports and dashboards differ?

- ☐ In reports, you can have multiple pages; in dashboards, you can have only one page.
- ☐ In reports, you can use the slicers and filter by selecting a data point on a visual; in dashboards, you can only filter a dashboard tile in focus mode, but can't save the filter.
- ☐ You can only build reports and dashboards in Power BI service.
- ☐ They are the same.

Question 3

Can a dashboard be created from multiple reports?

- ☐ No, dashboards can only be created from a single dataset or report.
- ☐ Yes, dashboards can be created from multiple datasets or reports.

Real-time Dashboards

Create a real-time dashboard

In this data-centric world, it has become increasingly important to have the ability to view how data changes in real time. This ability is particularly important in the context of dashboards; these are the canvases on which you can tell the story of the data, so the ability to show real-time, streaming data on these dashboards can be important to your business. With Power BI's real-time streaming capabilities, you can stream data and update dashboards as soon as the data is logged.

To continue with the module scenario, you are helping Tailwind Traders understand how well their manufacturing floor is operating. The assembly line has machines that are broadcasting a telemetry event each time that they perform their functions. You are collecting those event messages and want to display them with a Power BI visual. Dashboards allow you to use streaming datasets for this purpose.

Stream in Power BI

Streaming data can come from a variety of sources, including from social media, factory sensors, service-usage metrics, and other sources that contain a constant stream of data points.

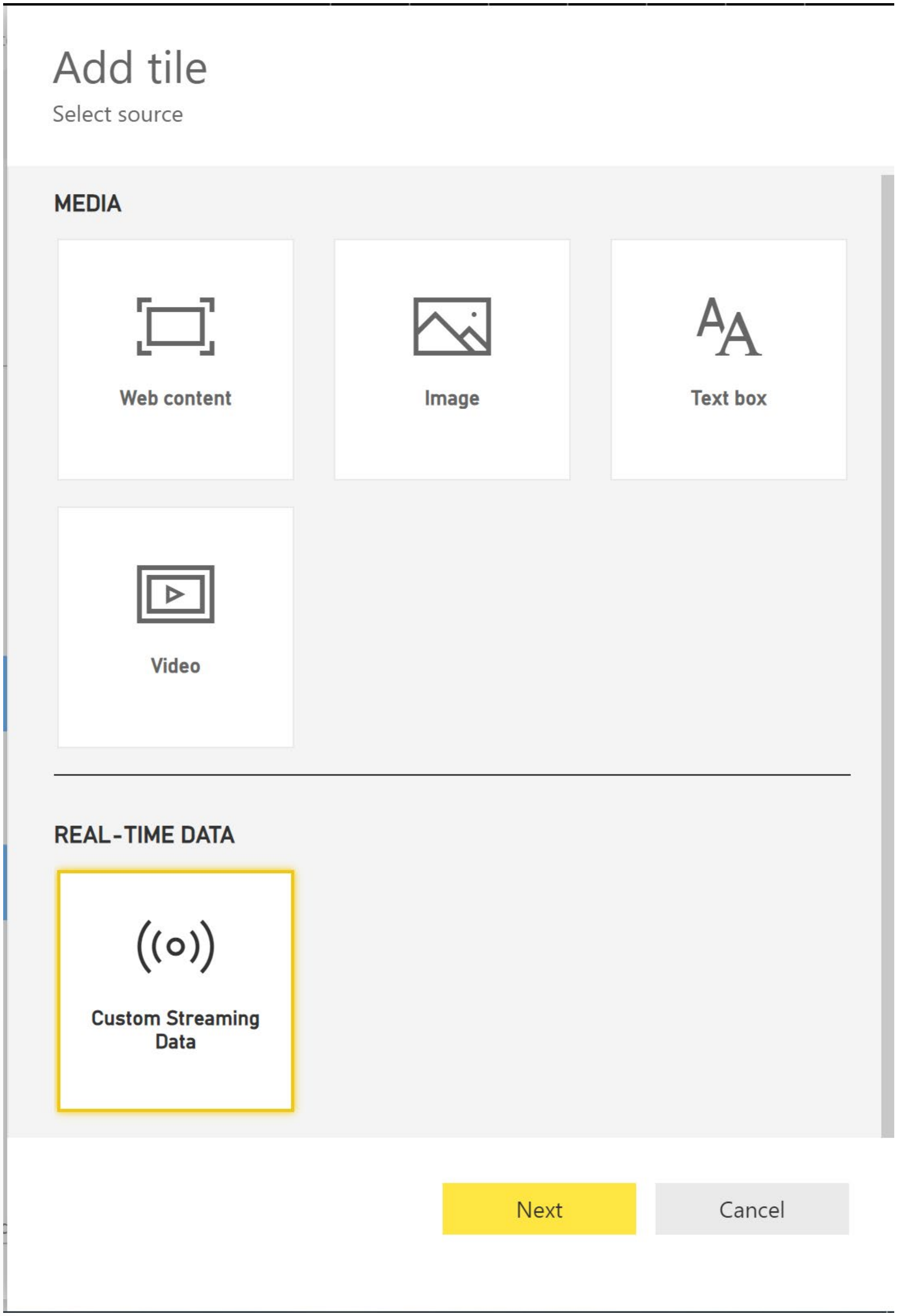
For instance, in the case with Tailwind Traders, sensors on the machines constantly send a stream of telemetry data to the IoT hub, where they will be housed in their native, messy format. From the IoT hub, you can use a stream insight job to aggregate the data, meaning that it will clean the data and quiet the noisy messages. Then, you can retrieve the data into Power BI as a streaming dataset, where you can consume the information and build the pertinent visuals.

Data that comes from a streaming dataset is not stored in a Power BI data model; instead, it is stored in a temporary cache. Consequently, you cannot model the data with this type of dataset. The only way to visualize the data from a streaming data source is to create a tile directly on a dashboard and use a custom streaming data source. These tiles are optimized for displaying the data quickly and, because no database exists to pull the data from, these types of tiles have low latency and are best suited for data that doesn't need additional transformations, such as temperature or humidity.

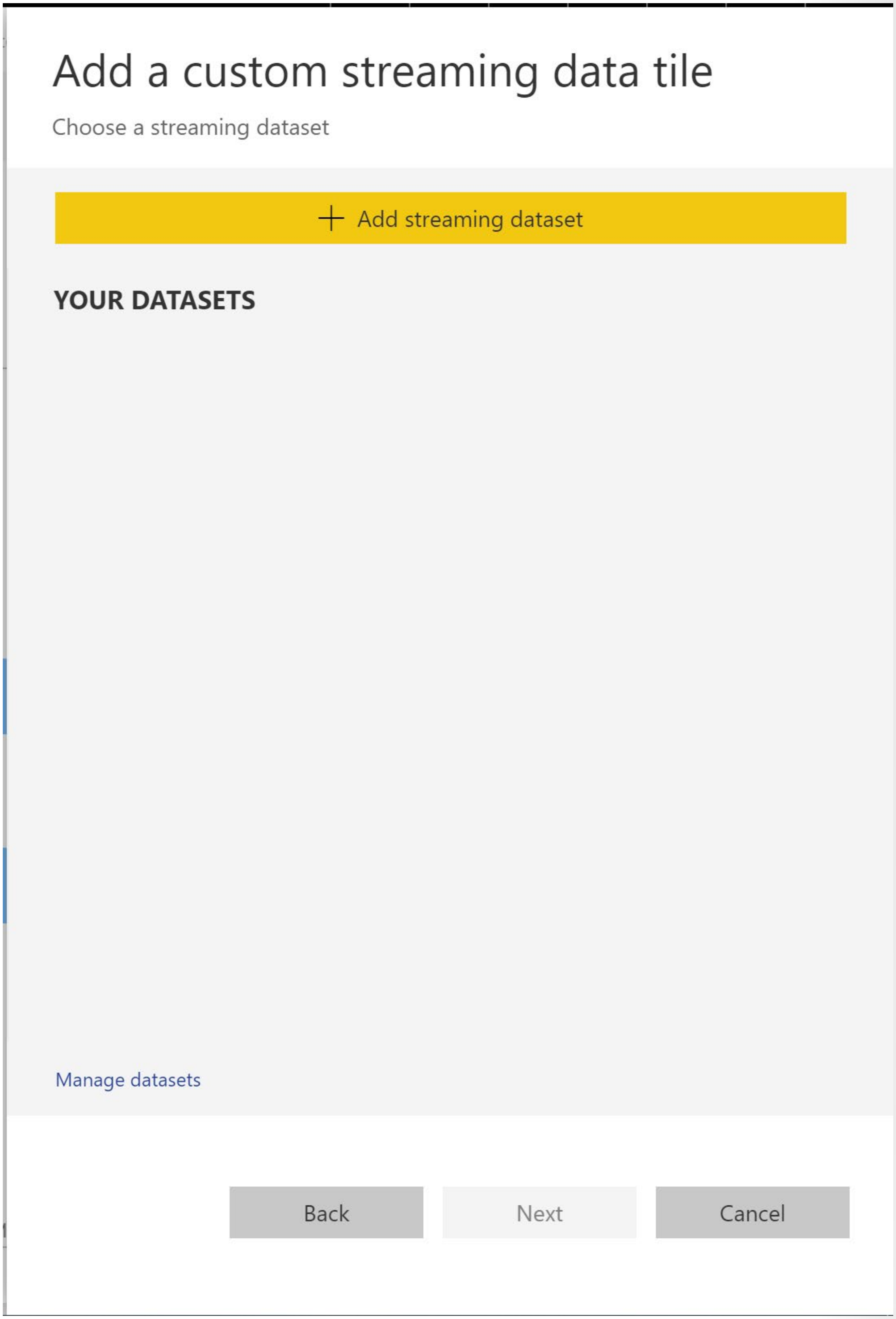
Visualize real-time data in Power BI

To visualize streaming data, you need to create a new tile directly on an existing or new dashboard.

To complete this task, go to and open a dashboard and then select **Add Tile**. The following window will appear, where you can select **Custom Streaming Datasets** under **Real-Time Data**.


Add tile
Select source
MEDIA
Web content
Image
Text box
Video
REAL-TIME DATA
Custom Streaming Data
Next
Cancel

Select **Next**, which will redirect you to the following window where you can choose an existing streaming dataset, or get new streaming datasets, as shown in the following image.



After you have selected the new dataset, select **Next**, enter the details for your streaming dataset, and then add a new streaming dataset tile. Streaming dataset tiles can be in the form of line charts, stacked bar charts, cards, and gauges and are formatted similarly to any other kind of tile.

For more information, see **Real-time streaming in Power BI**[3].

Pin a live report page to a dashboard

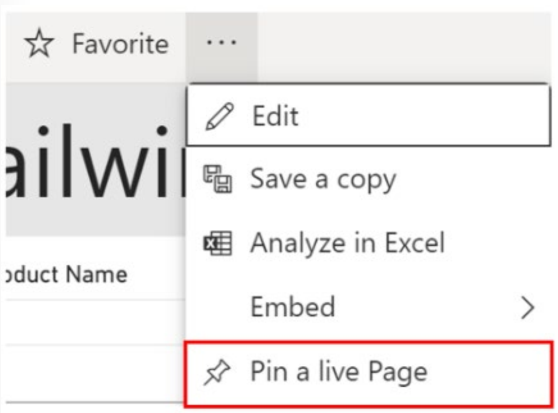
The process of building reports and dashboards is iterative. As data is constantly refreshed and business requirements change, it is expected that your reports and dashboards might also change; both in what filters or slicers you might have and also in what report elements, charts, and cards you have. For this reason, it is crucial that Power BI supports this iterative process. Through Power BI's innate functionality to pin live report pages to a dashboard, you can ensure that you aren't using old data and the visuals on your dashboards reflect changes live.

To continue the module scenario, you have built a few reports for Tailwind Toys. Several months go by, and the business requirements in the Sales team change, where they want you to change and add a few more visuals to the reports. When deploying your reports to Power BI service and creating dashboards, you want to ensure that you won't have to keep publishing new reports and dashboards every time a change occurs. You want to make sure that your changes are shown live. By using the pinning live reports to a dashboard feature from Power BI, you can complete this task in an intuitive manner.

Pin a live page

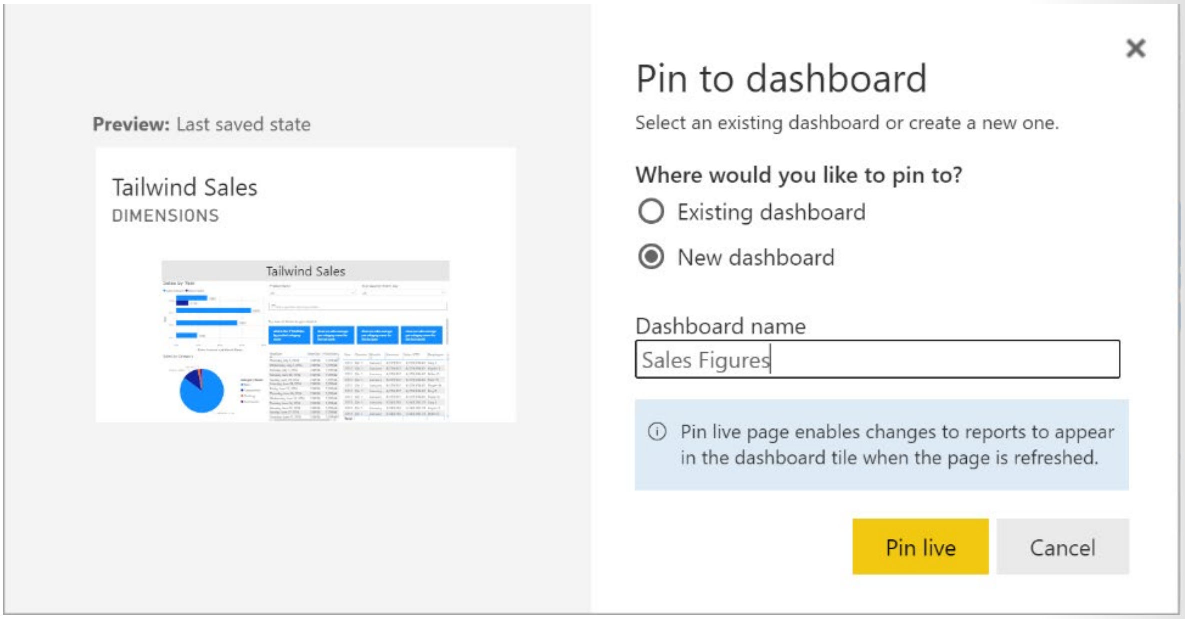
When you pin a visual, you can add it to a new or an existing dashboard. You can do the same with entire reports; when you pin a report page, all visuals on the report will be pinned to a dashboard and they are also live, meaning that any changes you make on the report will be immediately reflected on the dashboard that you have pinned the report to.

Pinning a live page is a simple way to pin all visuals at once so that you don't have to do any reformatting on the dashboard. To pin a live page, select the ellipsis (**...**) on the navigation bar of the report and then select **Pin a live Page**.

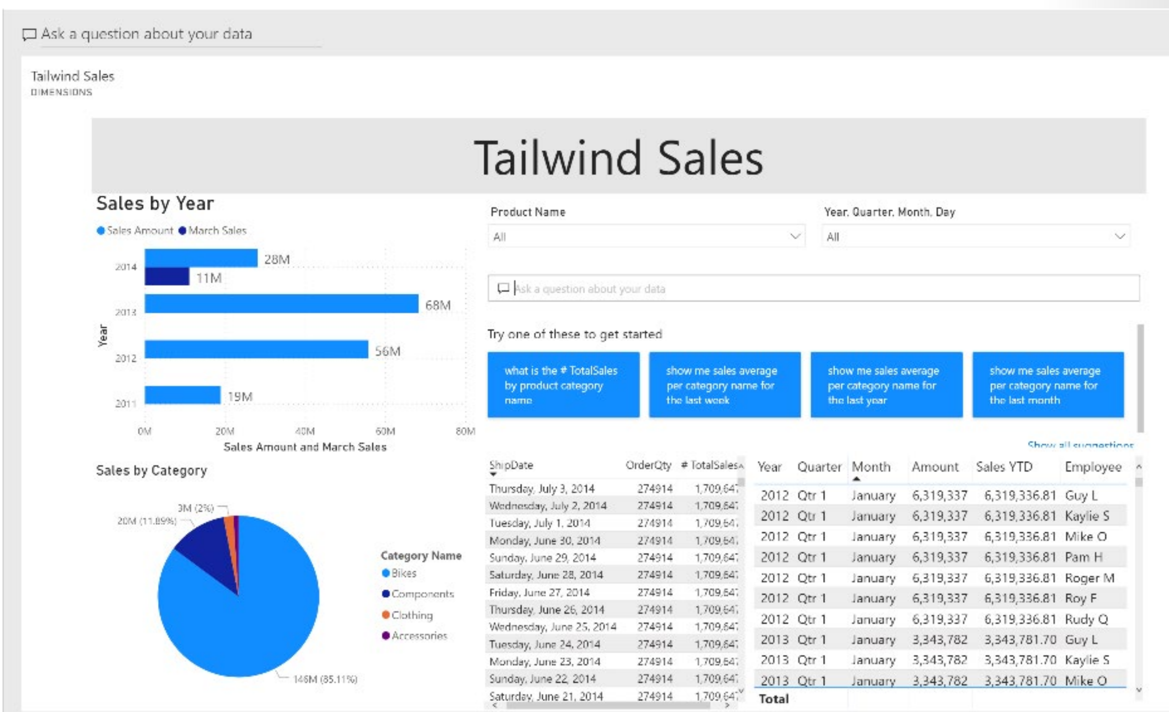


After you have made the selection, you can choose whether you want to pin this report to a new dashboard or an existing one. For this scenario, you want to pin your report to a new dashboard called **Sales Figures**.

3 https://docs.microsoft.com/power-bi/connect-data/service-real-time-streaming



After selecting **Pin live**, you will be redirected to a new window where you can see your dashboard. On the dashboard, you can modify the visuals as needed. Note that all your slicers and filters still work and that the visuals have the same data as in the report.



Any changes that you make to the tickets report will automatically show on the dashboard when the page is refreshed. In Power BI Desktop, you can make changes to your visuals or data as needed and then deploy to the appropriate workspace file, which will update the report and simultaneously update the dashboard as well.

You have now learned how to pin visuals as individual tiles and as entire live report pages. A word of caution: Dashboards are intended to be a collection from various sources, not just as a “launching pad” for reports. We recommend that you pin at the tile level first and foremost, and if needed, the entire report page can also be pinned. Seeing an entire report page in a dashboard tile can be difficult.

For more information, see **Pin an entire report page**[4].

Knowledge Check

Question 2

What is a key benefit of Power BI's real-time streaming capabilities?

- ☐ Users are limited to the data refresh as established by the developer.
- ☐ You can stream data and update dashboards as soon as the data is logged.

Question 3

Pinning an entire report page to a dashboard ensure what?

- ☐ Users are seeing individual tiles displaying key results.
- ☐ Pinning a page is an easy way to pin more than one visualization at a time. Also, when you pin an entire page, the tiles are live; you can interact with them right there on the dashboard.

4 https://docs.microsoft.com/power-bi/create-reports/service-dashboard-pin-live-tile-from-report

Enhance a Dashboard

Explore data by asking questions (The Q&A feature)

Power BI dashboards are about having a user-friendly experience. Dashboards in Power BI service are comprised of a canvas of interactive tiles, or report elements, that tell a data story.

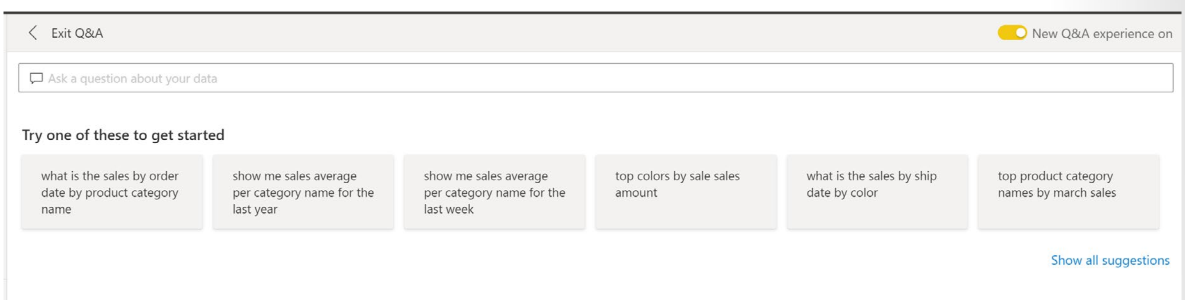
In this module's scenario, you are developing dashboards at Tailwind Traders. These dashboards are published; however, you begin receiving emails from users who are asking questions about the underlying data and are inquiring if you could build other visuals that are specific to their needs. A few questions might be manageable to answer, but situations might occur where you receive several emails and aren't able to keep up with demand. Power BI solves this problem with the Q&A visual. From the dashboard view, people can ask questions by using the **Ask a question about your data** search bar at the top of the dashboard, which increases engagement between users and the dashboard.

Q&A feature

The Q&A feature is a tool within Power BI Desktop that allows you to ask natural-language questions about the data.

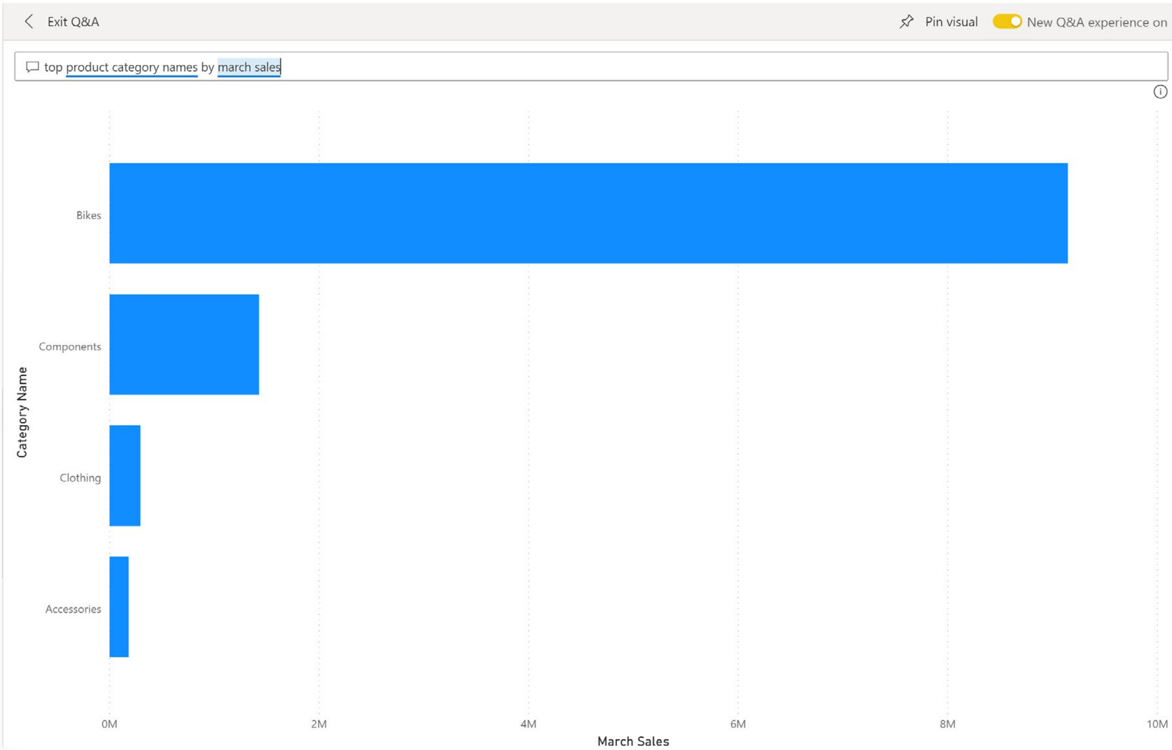
To locate the Q&A feature, go to your dashboard in Power BI service. Along the top ribbon is the **Ask a question about your data** search box.

After selecting this box, you will be routed to the following page.



The Q&A visual consists of three main elements:

- **Question box** - In this element, the user can enter their question about the data. An example of a question could be: What was the average sales amount by category? Entering this question will trigger Power BI's natural-language analysis engine to parse and determine the appropriate data to display.
- **Pre-populated suggestion tiles** - This element contains tiles with pre-populated suggestions for questions that the user can consider asking. When the user selects one of these tiles, they will be shown analysis. For example, if you select the **top product category named by march sales** tile, you would get the following visual that is converted from the Q&A visual.



- **Pin visual** icon - This icon is located in the upper right of the visual, as shown in the following image.

Selecting the **pin visual** icon will allow you to pin the visual onto a new or existing dashboard, as you have done previously.

With the Q&A feature, you can return to your users with a solution to their questions. Now, they can interact directly with the visual to ask their data questions, which will increase their interactions with the visual and help them save time.

Quick Insights

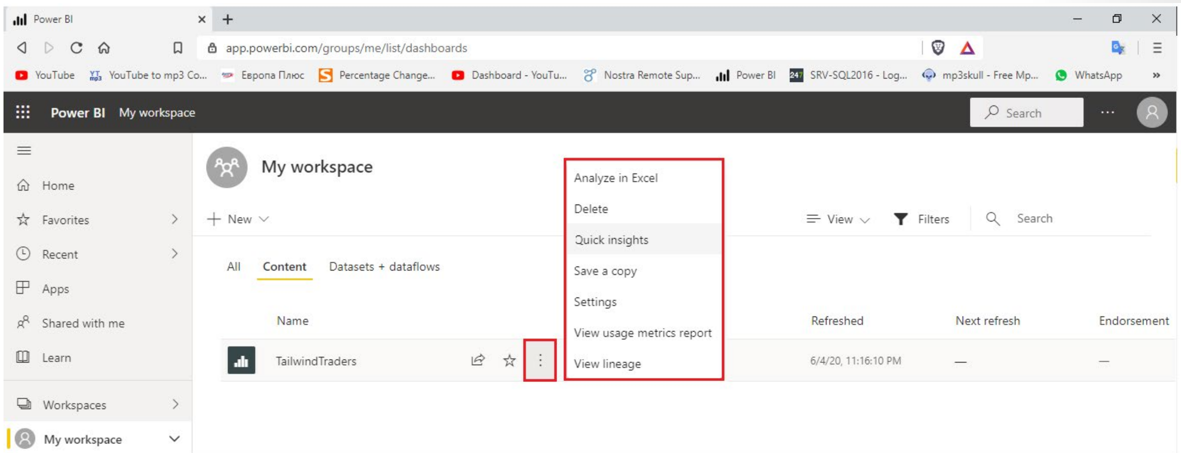
The **Quick insights** feature in Power BI uses machine learning algorithms to run over your entire dataset and produce insights (results) for you, very quickly. This feature is a great way to build dashboards, when you don't know where to start. It also helps you to determine any insights you might've missed when building out your reports. From the insights that Power BI discovers, you can generate interesting interactive visualizations.

NOTE: This feature is available in the Power BI Web Service only. Also, this feature doesn't work with DirectQuery; it only works with data imported to Power BI.

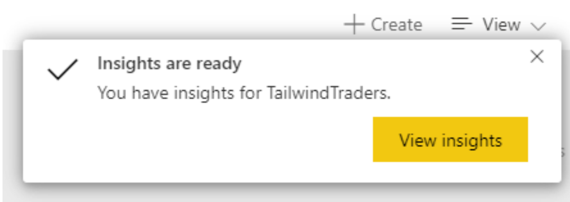
Suppose one of the datasets you've been given contains a massive amount of data concerning the Help tickets created for the Customer Service team. You don't know where to start analyzing, as there is just so much data, so you decide to let Power BI do it for you.

Get quick insights on your dataset

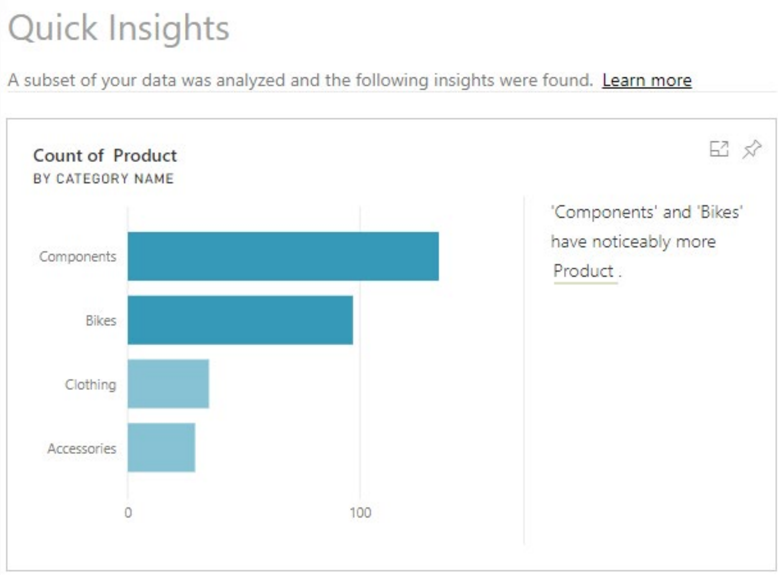
To get quick insights on your dataset, open your Power BI Web Service, then select the **Content** tab. Locate your report for which you want to get quick insights, in this case it is the **TailwindTraders**. Then select **More options** (...) > **Quick insights**.



Power BI will use various algorithms to search for trends in your dataset. This process might take a few seconds but when it is finished, you'll see a message in the top right corner letting you know that the results are ready to be viewed.



Select **View insights** to open the **Quick Insights** page for the selected dataset and view the insights that Power BI found for you. The **Quick Insights** page contains up to 32 separate insight cards, and each card has a chart or graph, plus a short description. In this example, one of the insights is a card that displays a visual for **Count of Products by Category Name**, as illustrated in the following image.

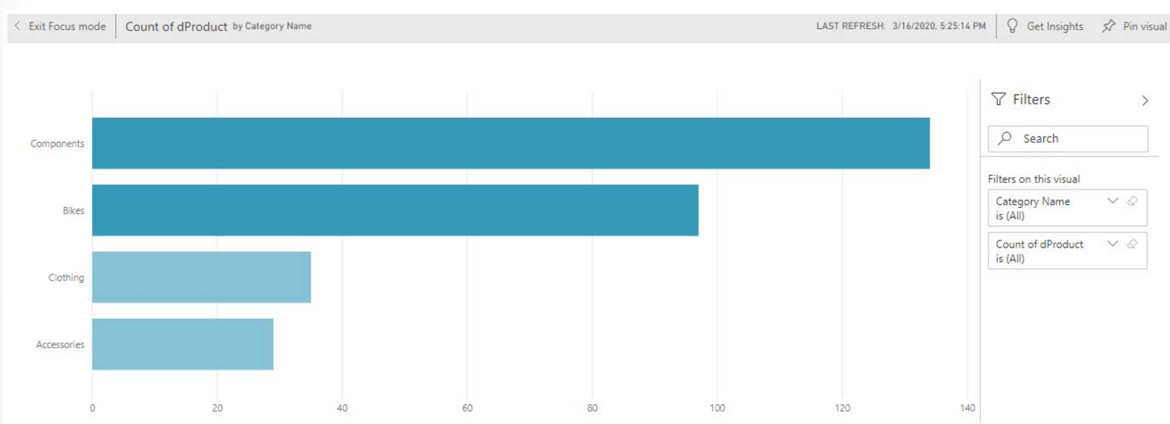


Add a quick insights result card to a report

If you see an insight card that is particularly interesting, you can add it to your report. On the **Quick Insights** page, hover over the card, then select the pin icon. The visual is added to your dashboard, where you can reposition it as required.

Interact with the quick insights results

To take a closer look at a particular insight card on the **Quick Insights** page, select an insight card to open the insight screen opens in **Focus** mode.



You can then perform the following actions:

- Filter the visualization using the available options in the **Filters** panel.
- Pin the insight card to a dashboard by selecting **Pin visual**.
- Run insights on the card itself (scoped insights) by selecting **Get insights** in the top-right corner. The scoped insights allow you to drill down deeper into your data.
- Return to the original insights canvas by selecting **Exit Focus mode** in the top-left corner.

Configure sensitivity labels

Power BI dashboards are an effective and visually pleasing way to disseminate information. They allow you to share business insights and concisely tell you the story of the data. However, because they can be seen by anyone who has been given access or a link, an important concern is security.

For instance, consider that you have built a few dashboards for the Sales team at your organization. You want to make sure that the users who have been given access know how the data within these dashboards is appropriately labeled. Microsoft Information Protection (MIP) sensitivity labels in the Power BI service allow you to complete this task.

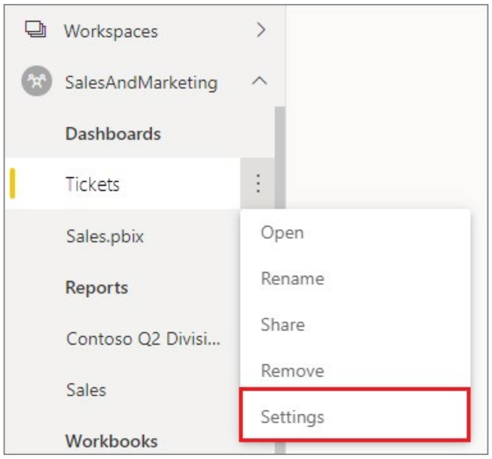
Apply sensitivity labels to dashboards

MIP sensitivity labels help the dashboard owner raise security awareness to viewers of a dashboard so that they know what level of security should be considered when viewing or sharing a dashboard.

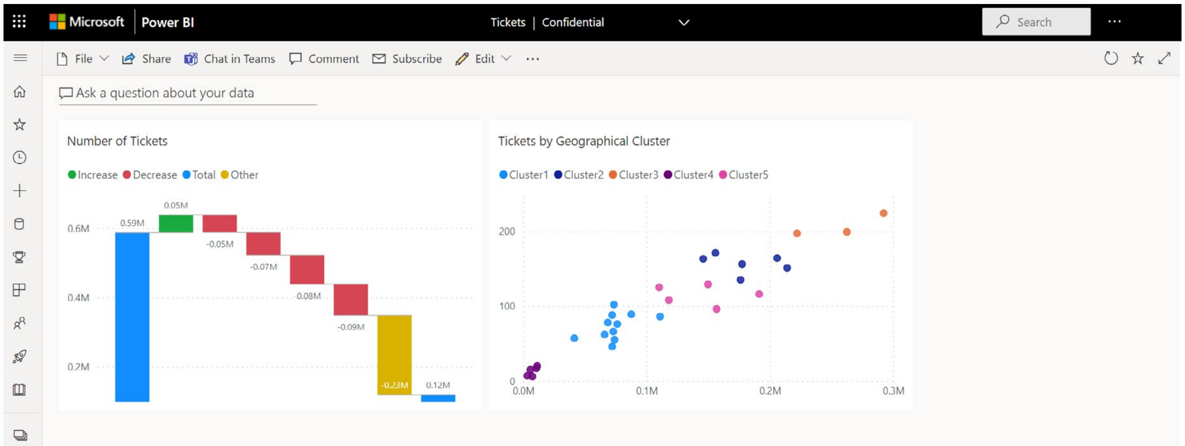
MIP sensitivity labels can be turned on and off in accordance with your organization's business needs. All dashboards are defaulted to a certain classification type; however, the dashboard owner can manually make changes to the classification.

To continue with the module scenario, you are working on the **Tickets** dashboard and want to add a sensitivity label to it. The first action that you will need to take is to ensure that your organization's custom data classification settings are added into the Power BI system. Defining sensitivity labels is done by an administrator.

To access MIP sensitivity labels on a dashboard, go to a specific workspace in Power BI service. Hover over the ellipsis (**...**) by the name of the dashboard and then select **Settings**.



When users open the dashboard, it will now be marked by this new sensitivity label, as shown in the following screenshot.



You have now added MIP sensitivity labels to your dashboards and the Sales team is pleased.

For more information, see **Dashboard data classification**[5].

Module Review

In this module, you have learned about dashboards: what they are, why you need them, and what tiles and pinning are in relation to dashboards. You have also learned how to accomplish several tasks around dashboards, such as:

- Setting mobile view.
- Adding a theme to the visuals in your dashboard.
- Configuring Microsoft Information Protection sensitivity labels.
- Adding real-time dataset visuals to your dashboards.
- Pinning a live report page to a dashboard.

With this new knowledge, consider how you can transform the data that you have to create a story. Dashboards can help you visualize that story.

For more information, see **Introduction to dashboards**[6].

Knowledge Check

Question 1

What feature allows you to ask a natural-language query about the data?

☐ The synonym feature

☐ The smart narrative visual

☐ The Q&A feature

5 https://docs.microsoft.com/power-bi/create-reports/service-data-classification
6 https://docs.microsoft.com/power-bi/create-reports/service-dashboards

Question 2

What feature in dashboards is used to alert consumers to the sensitivity of the data?

- ☐ Dashboard themes
- ☐ Microsoft Information Protection sensitivity labels

Answers

Question 1

What is a dashboard?

☐ A canvas of report elements that can be built in Power BI desktop.

☐ Dashboards can be built by using visuals that are developed with an underlying data source.

■ A canvas of report elements that can be built in the Power BI service.

☐ The canvas in which you can view the data model of a report.

Question 2

What is one way that reports and dashboards differ?

■ In reports, you can have multiple pages; in dashboards, you can have only one page.

☐ In reports, you can use the slicers and filter by selecting a data point on a visual; in dashboards, you can only filter a dashboard tile in focus mode, but can't save the filter.

☐ You can only build reports and dashboards in Power BI service.

☐ They are the same.

Question 3

Can a dashboard be created from multiple reports?

☐ No, dashboards can only be created from a single dataset or report.

■ Yes, dashboards can be created from multiple datasets or reports.

Question 2

What is a key benefit of Power BI's real-time streaming capabilities?

☐ Users are limited to the data refresh as established by the developer.

■ You can stream data and update dashboards as soon as the data is logged.

Question 3

Pinning an entire report page to a dashboard ensure what?

☐ Users are seeing individual tiles displaying key results.

■ Pinning a page is an easy way to pin more than one visualization at a time. Also, when you pin an entire page, the tiles are live; you can interact with them right there on the dashboard.

Question 1

What feature allows you to ask a natural-language query about the data?

☐ The synonym feature

☐ The smart narrative visual

■ The Q&A feature

Question 2

What feature in dashboards is used to alert consumers to the sensitivity of the data?

☐ Dashboard themes

■ Microsoft Information Protection sensitivity labels

Module 9 Perform Advanced Analytics

Advanced Analytics

Introduction to Advanced Analytics

Analytics encompasses emerging industry practices, such as data mining, big data analytics, machine learning, artificial intelligence (AI) and predictive analytics. It is a term used to describe the technical aspects of analytics that have predictive capabilities and can be used to solve business problems.

Analytics can transform raw data into an extensive collection of information that categorizes data to identify and analyze behavioral data and patterns. Organizations can use this information to not only analyze the current state of their operations but also to predict future behavior and trends, by asking "what-if" questions. But that's not all it can do. It can help with fraud detection, image recognition, sentiment analysis, and overall general employee productivity. Also, it often replaces cumbersome manual processes.

Imagine asking an employee to figure out what is causing a recent spike in sales. The employee might have to painstakingly comb over each sale, interview customers, talk to sales people, and examine market trends. Instead, you can use the Power BI key influencers visual to use advanced analytics and possibly get to an answer much faster. The visual is only as good as the data you give it, so you'll still have to collect the data and organize it. The actual analytics, however, can easily be done for you, or at least give you an excellent start.

Advanced analytics ultimately enables organizations to make better business decisions and create actionable and meaningful results, by reducing manual work.

Traditionally, data analysis was a complex task carried out by engineers but today, data analysis is more accessible to, and understood by, many people within organizations, across all teams. Power BI is a perfect tool for quickly pulling actionable insights from data. It allows you to build visuals and metrics for your data in reports and dashboards, so that you and the end users can analyze data insights at a high level, and drill down into those insights for more detailed information.

In this module's scenario, you work for Tailwind Traders as a data analyst. You've been tasked with building reports and dashboards that will be used across the organization to aid crucial business decisions. For example, the Product team is interested in learning if there are specific products that are not selling as well as others, the Sales team is focused on sales forecasts for the coming year, and the warehouse team is interested in a general breakdown of the how the warehousing and shipping locations are performing worldwide. For each of these teams, you have to build and share unique reports and dashboards that display high-level insights, as well as visuals developed using advanced analytics.

Power BI's inherent functionality will make it easy for you to tackle this task. You can develop quick insights and share them easily in reports and dashboards with different teams within the organization. The advanced analytics capabilities of Power BI will enable you to identify categories and trends, see how data changes over time, and much more. From this information, you can make predictive data models, and therefore, help your organization to make more robust business decisions, plans, and forecasts.

This module outlines the advanced analytic capabilities of Power BI. By the end of this module you will be able to:

- Identify outliers with Power BI visuals
- Use the Analyze feature
- Use advanced analytics custom visuals
- Review Quick Insights
- Group and binning data for analysis
- Apply clustering techniques

Identify outliers

An outlier is a type of anomaly in your data - something that you didn't expect or that surprised you, based on historical averages or results. You will want to identify outliers to isolate data points that significantly differ from other data points, then take action to investigate the reasons for the differences. The results of this analysis can make a huge impact on business decision making.

Imagine you are analyzing data for a shipping warehouse. You notice that number of orders spiked up above average for specific product category. You first want to identify which product category it is. Then, you want to ask several questions about the outlier. Was there above average shipments that day? Was it just a specific warehouse? Did a single event cause the spike in orders for specific category? Were there any other days like this in the last month, quarter, year, or prior year?

Power BI allows you to easily identify outliers in your data but you need to first determine the logic behind what constitutes an outlier. You can use trigger points, such as calculations, around what you would consider the outlier to be.

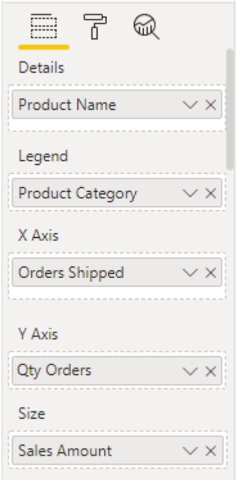
The process of identifying outliers involves segmenting your data into two groups; one group is the outlier data and the other group is not. You could use calculated columns to identify outliers but the results would be static until you refresh the data. A better way to identify outliers is to use a visualization or DAX formula, as these methods will ensure your results are dynamic.

When you have identified the outliers in your data, you can then use slicers or filters to highlight those outliers, and add a legend to your visuals, so the outliers can be easily identified amongst the other data. You can then drill in to the outlier data for more detailed analysis.

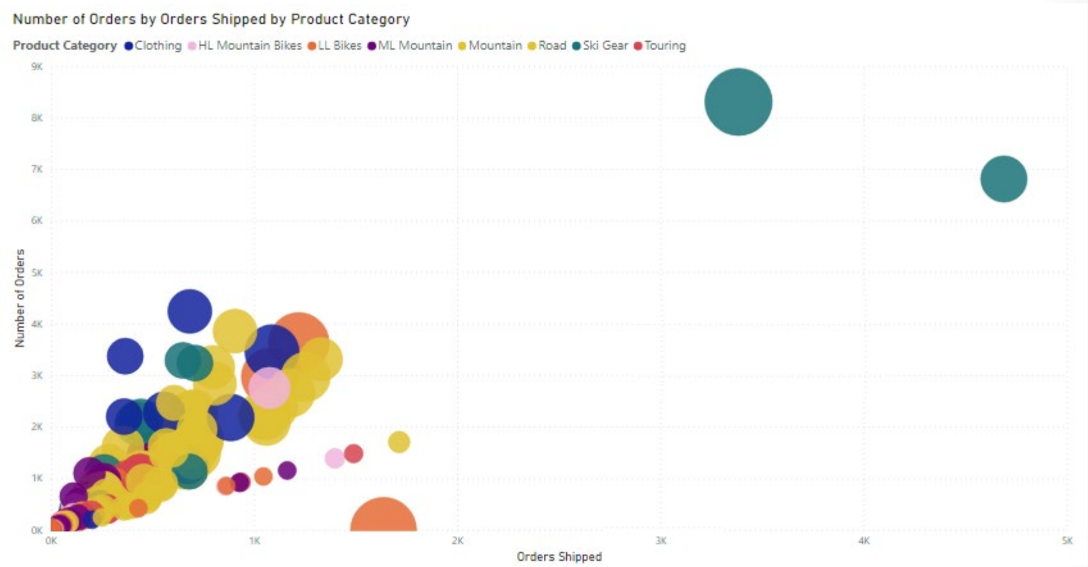
Use a visual to identify outliers

The best visual to use to identify outliers is the scatter chart, which shows the relationship between two numerical values. Scatter charts display patterns in large sets of data and are therefore, ideal for displaying outliers.

When you add a scatter chart to your Power BI report, you put the interesting fields in the **X axis** and **Y axis** sections respectively. In our case, the **Orders Shipped** field is on the X axis and the **Qty Orders** field on the Y axis.



The visual will update to display the data according to the selected fields, and you'll be able to clearly spot the outliers in that data – they are the isolated items that are away from the bulk of the data.



Now that you can identify the outliers in your data, you can investigate the reasons for their existence and take corrective action.

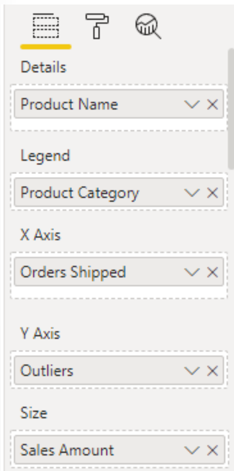
Use DAX to identify outliers

You can use DAX to create a measure that will identify the outliers in your data, such as in the following formula:

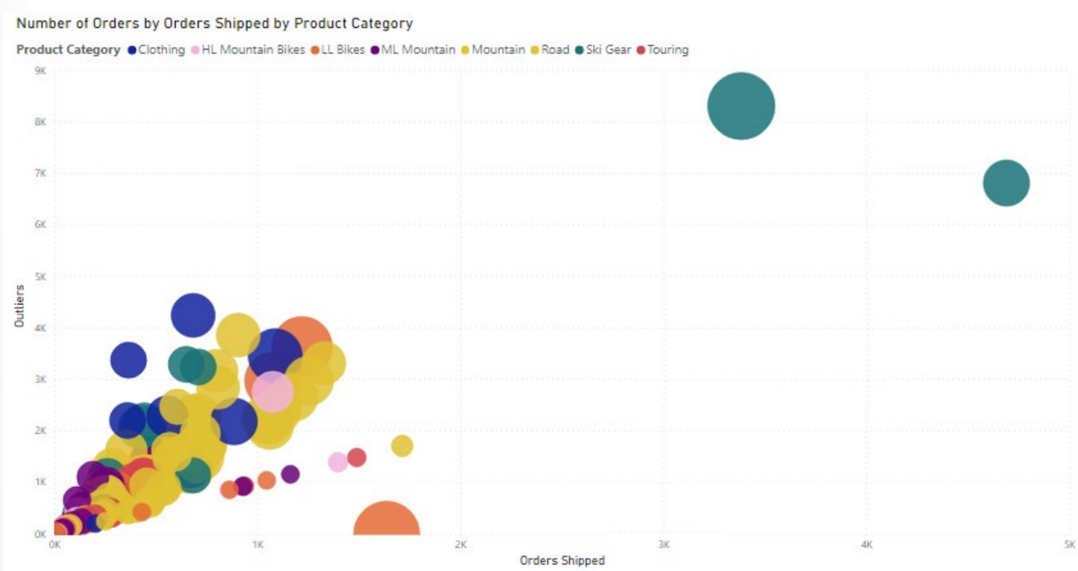
```
Outliers =
CALCULATE (
    [Order Qty] ),
    FILTER (
        VALUES ( Product[Product Name] ),
```

```
            COUNTROWS ( FILTER ( Sales, [Order Qty] >= [Min Qty] ) ) > 0
        )
    )
```

When you have created a new outlier measure, you can group your products into categories using the grouping feature, as you did when creating a histogram earlier. You then need to add a scatter chart visual, as you did in the previous section, as this the best visualization option for displaying outliers. When you've added the scatter chart, you populate it with the fields associated with your DAX formula and outlier measure.



Again, in the scatter chart, you'll be able to identify the outliers in your data. You can then investigate the reasons for their existence and take corrective action.



Using the Analyze feature

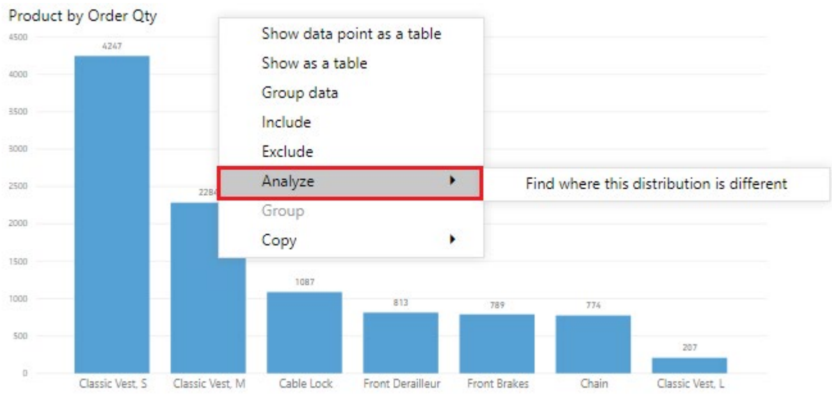
The **Analyze** feature provides you with additional analysis that is generated by Power BI for a selected data point. You might want to use this feature to see if Power BI has found something that you haven't looked at before, or if you want Power BI to give you a different insight to your data. This feature is particularly useful for analyzing why your data distribution looks the way it does.

NOTE: This feature does not work if you have non-numeric filters applied to your visual and/or if you have measure filters applied.

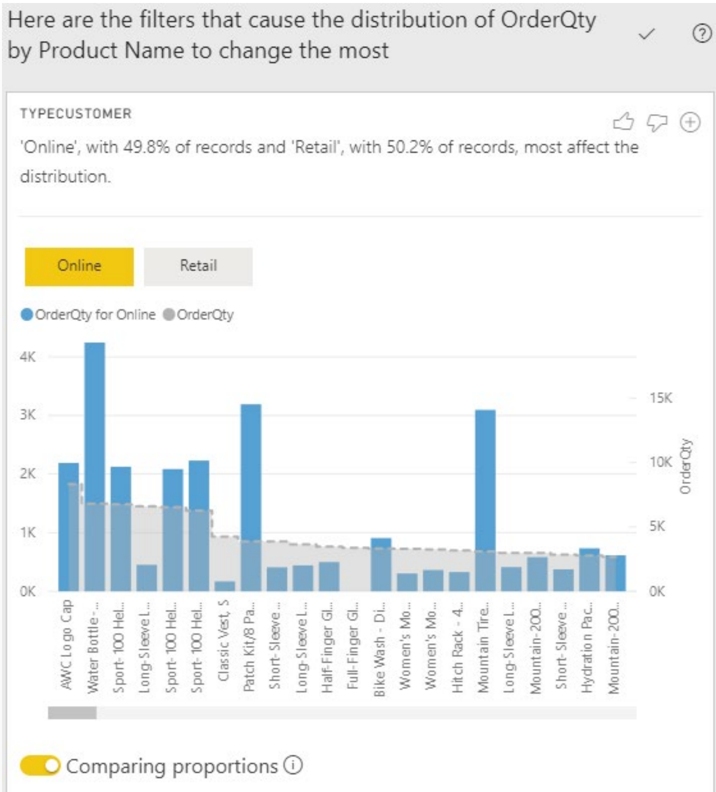
In this example, you are developing a report for the Customer Service team that deals with Help tickets. They want to analyze the ticketing data that is created online when a customer asks a question. You've created a preliminary visual to display data for tickets by location but you're now curious as to why the distribution of your data looks the way it does.

Instead of exploring the data manually, you can use the **Analyze** feature to get a fast, automated, insightful analysis on your data.

To use the **Analyze** feature, right-click a data point on the visual, then hover over the **Analyze** option to display two further options: **Explain the increase** and **Find where the distribution is different**. The options that are available will depend on the data point you selected.



In this example, you select the **Explain the increase** option and a window displays with a new visual, as illustrated in the following image.



If you find this analysis useful, you can add the new visual to your report, so other users can view it. Select the plus icon in the top right corner of the visual to add it to your report.

To learn more about the Analyze feature, see **Apply insights in Power BI Desktop to discover where distributions vary (preview)**[1].

Advanced analytics custom visuals

In addition to the out-of-the-box visualizations you see in Power BI Desktop, the Microsoft AppSource has a vast library of custom visuals that you can import into Power BI Desktop. These custom visuals give you a wider choice of options when it comes to using advanced analytics. There might be a custom visual that solves a business problem that the standard visuals cannot solve, or one that presents your data in a way that the standard visuals cannot.

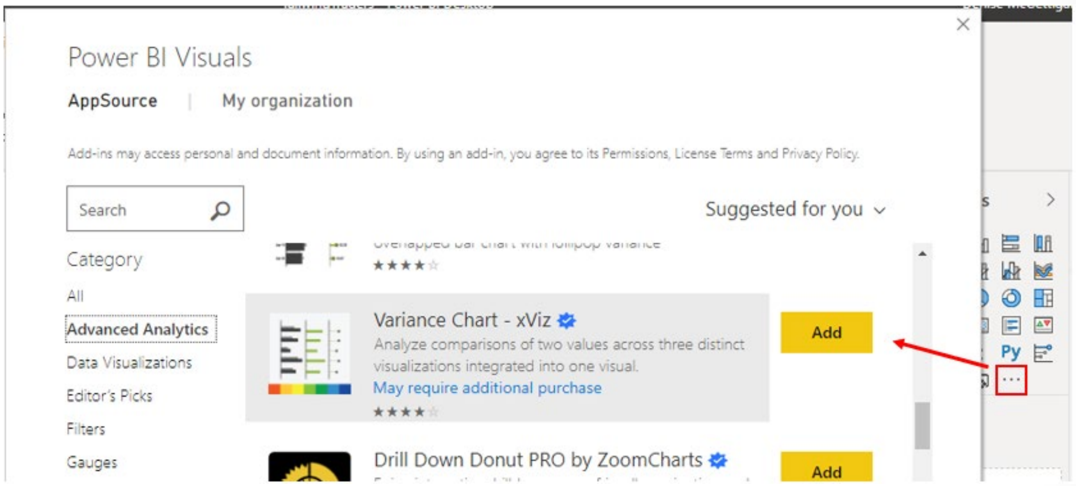
NOTE: Some organizations prefer not to use custom visuals at all, or only permit certain custom visuals, for security or other reasons. Before you import any custom visuals, check with your organization to see if they are allowed or not. If they are not allowed, you can still create reports in desktop with them, but they will not render in the Power BI Service.

You already imported a custom visual from the Microsoft AppSource in an earlier unit, for the purposes of time series analysis. In this unit, you'll focus on the range of Advanced Analytics custom visuals that are available. These custom visuals include box-and-whisker plots, variance charts, hierarchical trees, Gantt plots, clustering plots, and much more; the list goes on. Using advanced analytics visuals adds a layer of complexity to your reports, and allows you to further analyze the data and develop granularity within your visuals.

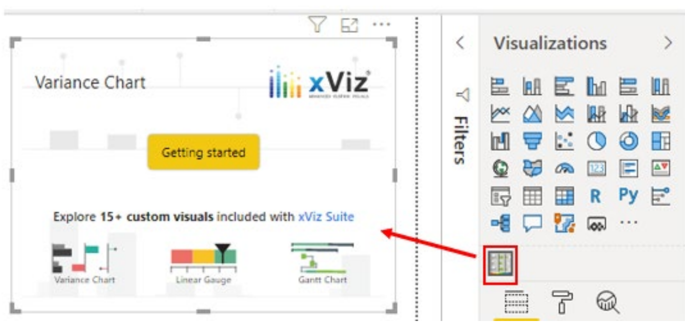
1 https://docs.microsoft.com/power-bi/create-reports/desktop-insights-find-where-different

In this example, you've produced some charts and visuals for the Customer Service team but now they want you to create a variance chart, so they can study the variance in the Help tickets. You decide to browse the Microsoft AppSource to see if there is an advanced analytics visual you can use to satisfy this request.

In the **Visualizations** pane, select the **Get more visuals** icon, then select **Get more visuals**. On the **Power BI Visuals** window that displays, select the **Advanced Analytics** category. Browse the available options, then select the **Add** button for the visual you want to import. In this example, you add the **Variance Chart** custom visual.

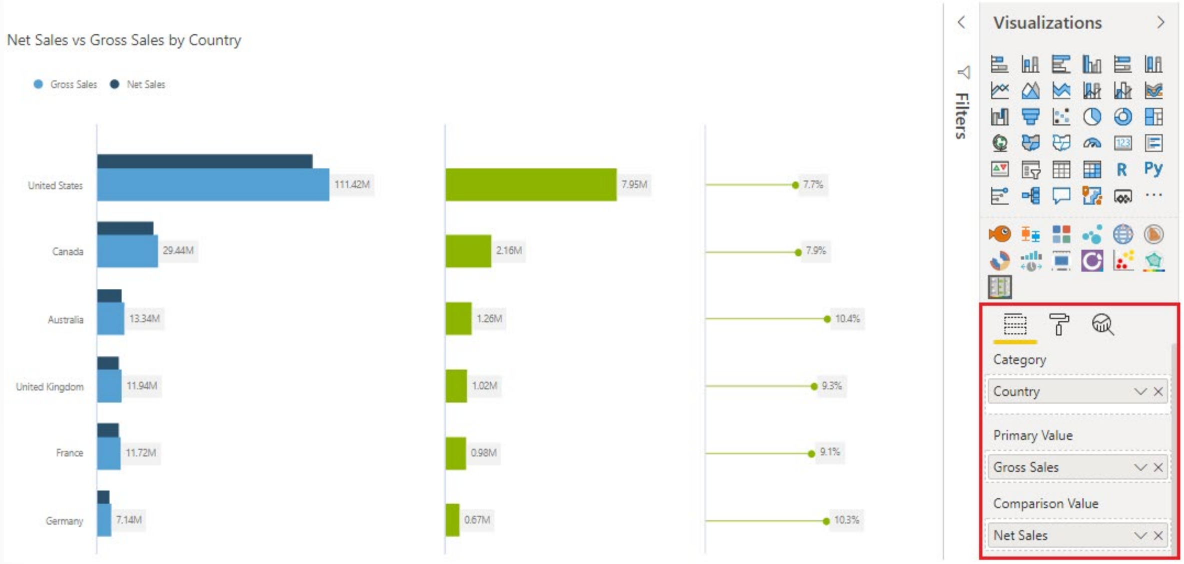


When imported, an icon for the new custom visual displays under the other visual icons in the **Visualizations** pane.



You can then add fields to the new visual and customize the visual, in the same way you would for any other visual. In this example, you add **Country** to the **Category** field, add **Gross Sales** to the **Primary Value** field, and add **Net Sales** to the **Comparison Value** field.

You then see that you have a variance visual that contains multiple charts, which is something you couldn't do without importing the Advanced Analytics custom visual from the AppSource.



Groupings and Binnings

When you create visuals, Power BI Desktop aggregates your data into groups, based on the values it finds in the underlying data. You can refine how those default groups are presented. You can also create new groups by grouping two or more data points in a visual or putting values into equal sized groups (binning).

Grouping is used for categories of data. Binning is similar to grouping but it is used for grouping continuous fields, such as numbers and dates.

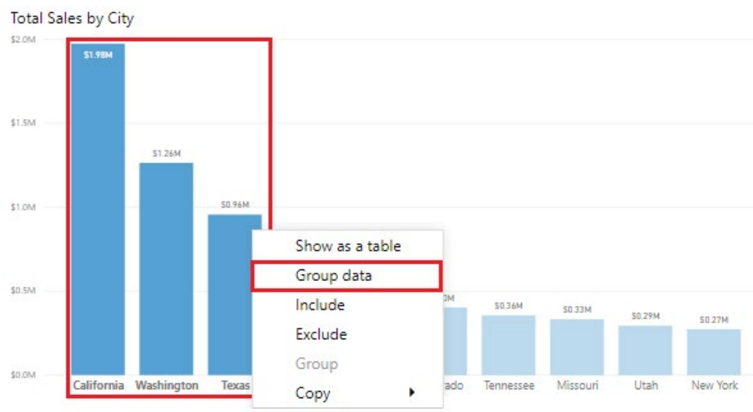
You can use the grouping and binning features to ensure the visuals in your reports display your data exactly how you want them to, so you can more clearly view, analyze, and explore the data and trends in your visuals. You'll be able to identify clusters, patterns of behavior, data averages, and more. The results of this analysis will provide your end-users with more specific insights on their data, which can drive business decisions.

In this example, the Customer Service team has come back to you, greatly impressed by the analysis you have done so far. They now want you to further analyze their Help ticket data – they ask if you can segment the data into different groups and clusters. In particular, they want to identify the cities with the highest sales.

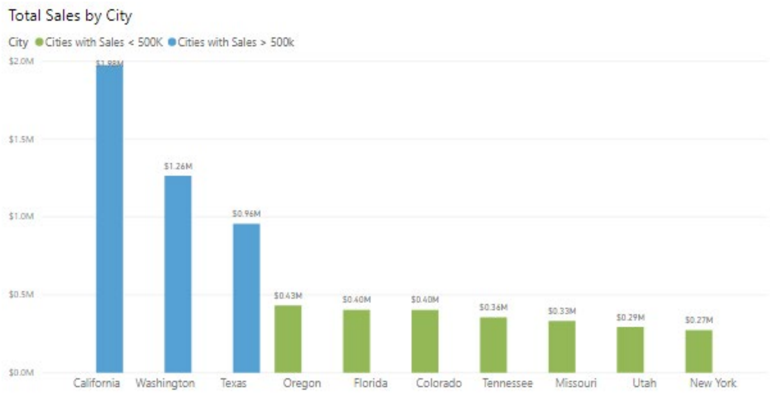
Create a group

In the following image, you can see a bar chart in which Power BI has automatically segmented the data in the way that it found most useful - Total Sales by City. However, you want to group some of the bars (cities) together, so you can view them as one category, which will help the Sales team to identify the cities with the highest sales.

To create the group, you start by using **Ctrl+click** to select the data points on the visual that you want to group, in this case it is cities with sales greater than 500 thousand dollars. You then right-click one of those selected data points and select the **Group data** option.



When the group is created, you'll see that the visual updates to take account of the new group. In the following image, you can see that the other cities, which are the cities with lower sales (less than 500 thousand dollars) have been grouped together and are highlighted in a different shade.



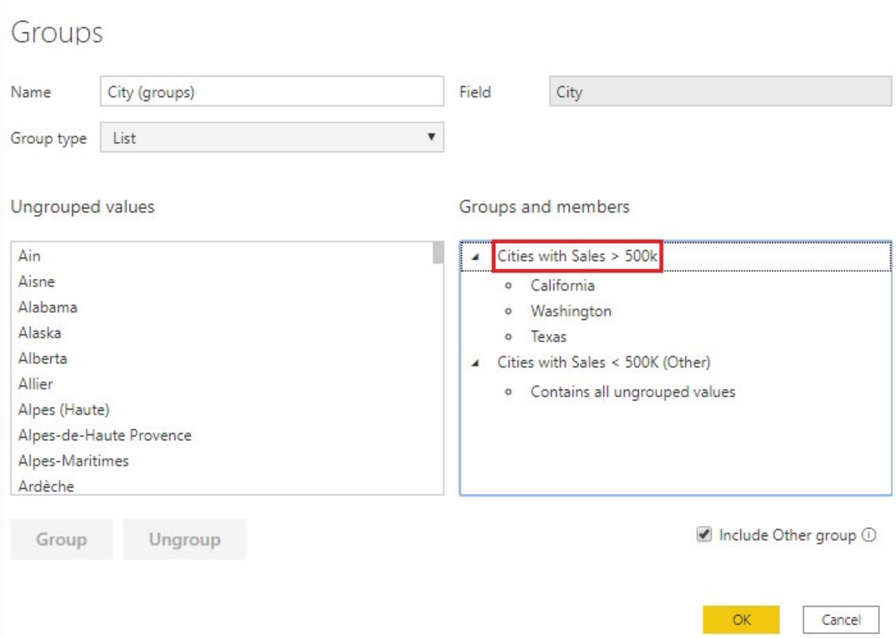
You'll see that the new group field displays in the **Legend** bucket for the visual and is listed in the **Fields** pane.

When you create a group, you can change the way the data is displayed in the visual. For example, you might want to switch the values in each axis. You can also use the group in any of the other visuals in your report. To do so, simply drag the group field from the **Fields** pane, then drop it into the visual in which you want to use it.

Edit a group

Continuing the above example, you now want to edit the categories that make up your group. Right-click the group field in either the **Legend** bucket or the **Fields** pane, and then select **Edit Groups**.

In the **Groups** window that displays, you can see a list of the groups and the different items within those groups. In the following image, you can see the **Cities with Sales > 500k** group and its members, along with the **Other** group (**Cities with Sales < 500k**) that contains all of the other values that have not been put into the first group. If you refresh your data and new items appear in the ungrouped values list, they'll all go into the **Other** group.

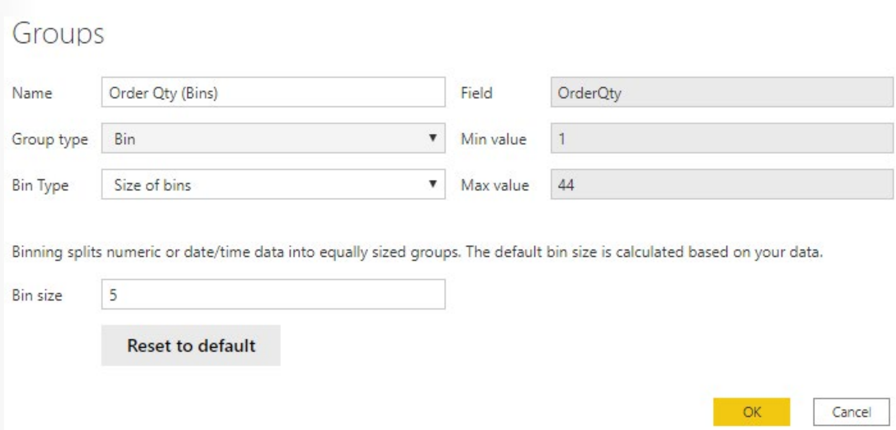


You can now make changes to the group. You can rename any group by double-clicking the group title in the **Groups and members** section and entering a new name. You can add ungrouped values into an existing group, remove values from an existing group, and even create a new group.

Create bin groups

The process of binning allows you to group your numerical and time field data into "bins" of equal size, so you can visualize and identify trends in your data in more meaningful ways. Binning allows you to right-size the data that Power BI Desktop displays.

In this example, you want to create bins (groups) for the **Order Qty** field. You start in the **Fields** pane, where you right-click the field **Order Qty** that you want to create the bins for, then select **New Group**. On the **Groups** window that displays, set the **Bin size** to the size you want, and adjust any other settings as required, then select **OK**.



When you have set up the bin group, you'll see a new field in the **Fields** pane with **(bins)** appended to its name. You can then drag that field onto the canvas to use the bin size in a visual.

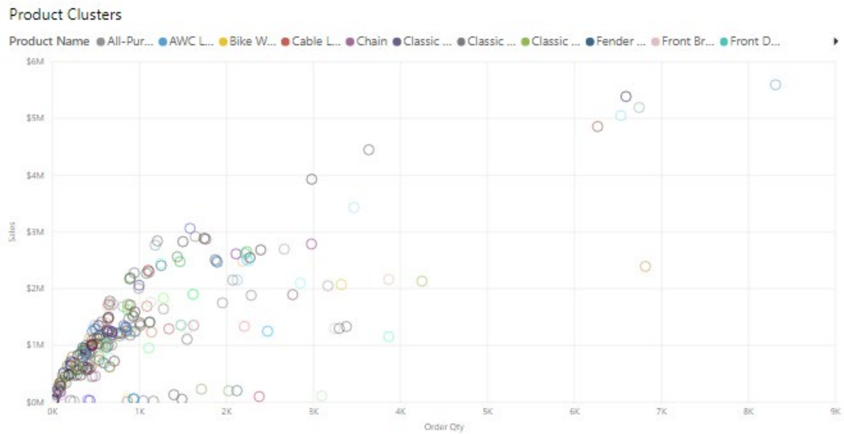
Clustering Techniques

Clustering allows you to identify a segment (cluster) of data that is similar to each other but very dissimilar to the rest of the data. The process of clustering is different to that of grouping, which you did earlier.

The Power BI clustering feature allows you to quickly find groups of similar data points in a subset of your data. It analyzes your dataset to identify similarities and dissimilarities in the attribute values, then it separates the data that has similarities into a subset of the data. These subsets of data are referred to as clusters.

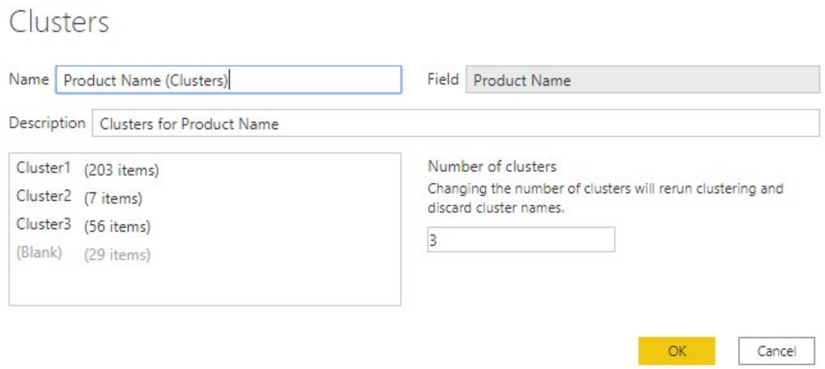
For example, you might want to look for patterns in your sales data, such as the behavior of customers overall. You can segment the customers into clusters, according to their similarities, such as age or location.

Start by adding the **Scatter chart** visualization to your report, then add the required fields to the visual. In this example, you add the **Order Qty** field to the X-axis, the **Sales** field to the Y-axis, and the **Product Name** field to the **Legend** section. As you can see in the following image, there is a lot of data now in the scatter chart, so it is difficult to see any natural groups.

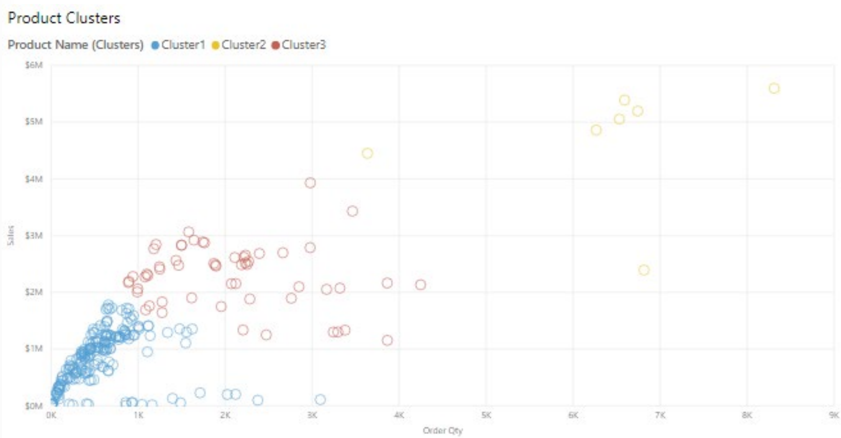


To apply clustering to your scatter chart, select **More options** (...) in the top right corner of the visual, and then select **Automatically find clusters**.

On the **Clusters** window that displays, you can edit the default name, field and description, if required. But in this example, you want to change the number of clusters. As you can see in the following image, the **Number of clusters** box is blank by default, which means Power BI automatically finds the number of clusters it thinks makes the most sense with your data.

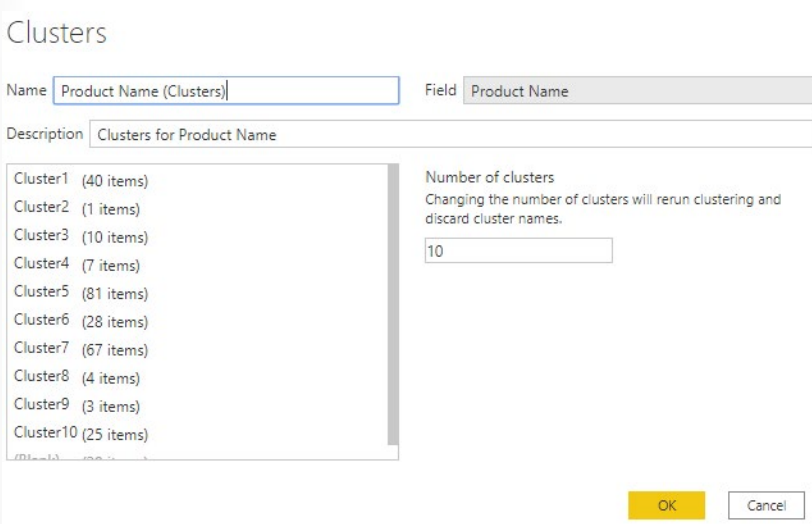


Enter the number of clusters you want (3) into the box, then select **OK**. Power BI will run the clustering algorithm and create a new categorical field with the different cluster groups in it. Now when you look at the visual, you can more easily see the clusters that are in your data, and proceed to perform analysis on them.



The new cluster field is added to your scatter chart's legend field well bucket, which you can now use as a source of cross highlighting like any other legend field. You can also find it in your field list and use it in new visuals, just like any other field.

If you want to edit the cluster, right-click the cluster field and select **Edit clusters**.



In the above example, when you applied clustering to the scatter chart, you could only use two measures. If you want to find clusters using more than two measures, you can use a table visual instead, add all the fields you want to use, then run the clustering algorithm using the same process.

Lesson Review

In this module, you learned about the emerging power of advanced analytics and discovered how the advanced analytics capabilities of Power BI can bring value to your organization's decision making and strategy formulation.

You enhanced the reports and dashboard you previously created, to enable end users across your organization to ask more questions and find even more information specific to their teams and work contexts. You used Power BI to explore your data and create a statistical summary of your data to find key takeaways and trends, and identify outliers that might require immediate action. You conducted a time series analysis of your data to see how your data progressed over time, so that you could make observations and forecast behavior. You used the Analyze feature to get Power BI to explain the increases, and show you where the distribution was different, in your data. You built upon the standard visuals in your report by adding custom visuals that helped you to solve additional business problems and display data in more enhanced, animated ways. You went back to Power BI to check for quick insights in your data that you had not already discovered and find more useful insights you could add to your dashboards. Lastly, you used the grouping, binning and clustering techniques to segment and present your data in different ways.

The advanced analytical techniques you applied to your organization's data enabled you to gain new insights in that data, and dive deeper into the data to uncover patters, trends, and outliers that you did not know existed. The results of your advanced analysis will empower your organization to make more robust business decisions, plans, and forecasts.

Knowledge Check

Question 1

What Power BI feature can give an in-depth analysis of the distribution of data?

- ☐ The Next Level of Hierarchy feature can give in-depth analysis because it will allow you to drill down for all subcategories and is not used to analyze the distribution.
- ☐ The Analyze feature allows a user to understand why the distribution of data looks the way it does.
- ☐ Only time series analysis can provide in-depth analysis on the data.

Question 3

What visual should be used to display outliers?

- ☐ The line chart is best-suited to display outliers.
- ☐ The clustered column chart is best-suited to display outliers.
- ☐ The scatter chart displays outliers.

Data Insights Through AI Visuals

Introduction

Microsoft Power BI gives you a wide range of methods to analyze your data and get answers to your questions. This module examines the Power BI features that are based in advanced analytics and visualization.

Occasionally, the fastest way to get an answer is to ask a question by using your own words. Power BI allows you to do exactly that method with its advanced analytics AI capabilities. One feature of Power BI is that it allows you to ask questions by using natural language, and then it will answer those questions for you.

Similarly, your dataset is likely numerical, where it consists of numbers, amounts, measures, and so on. You've been successfully analyzing this numerical data to get insights. Analyzing non-numerical data can be difficult, but with the AI capabilities of Power BI, you can analyze text data to get more insights than before. For example, you might have an abundance of comments or reviews from customers or the results of an employee survey that have been left long forgotten in a database. Now, you can analyze this additional data and transform it into valuable information.

Because the numbers don't tell you everything, you can use the AI features in Power BI to examine your data further, to see what people are saying, and to get more constructive and meaningful results, which will help you ultimately make better business decisions.

For this module's scenario, you have been building reports for different teams in your organization. Now, you want to use the AI aspect of the advanced analytic capabilities of Power BI to enhance your reports in clever ways. You want to empower users to gain new insights in more interactive ways and receive direct answers to their questions.

This module describes the AI capabilities of Power BI. By the end of this module, you will be able to:

- Use the **Q&A** visual.
- Find important factors with the **Key influencers** visual.
- Use the **Decomposition Tree** visual to break down a measure.

The Q&A Visual

The **Q&A** feature in Power BI lets you explore your data in your own words, by allowing you to ask natural language questions, and then answering those questions for you.

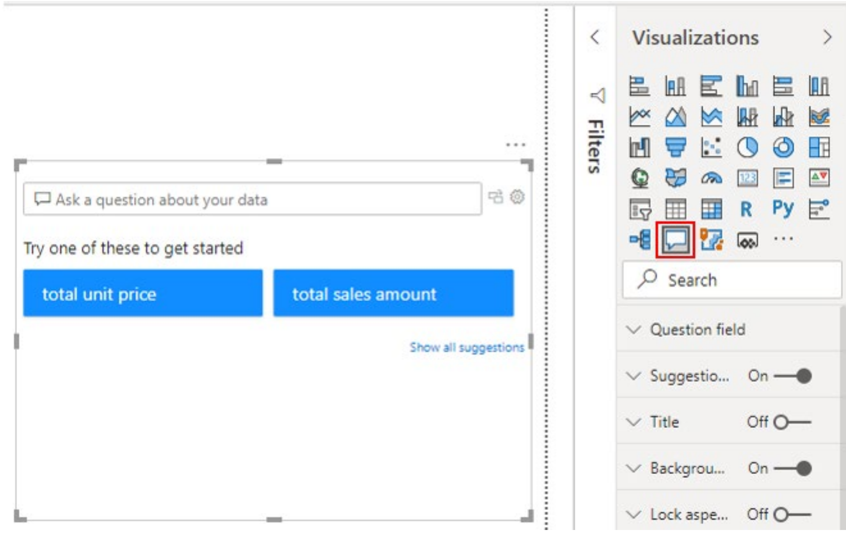
This ability to ask questions is valuable to both you, as the report author, and your report users. It gives you ideas for the type of visuals you can display in your report, and lets you quickly add those visuals. It gives your report users an effective tool they can use to get quick answers to their questions about the data, independently. This self-help aspect to Power BI saves time for everyone involved.

Power BI records all of the questions that are asked, and you can use this information to set up the **Q&A** feature to be more effective. When the **Q&A** feature answers so many questions, you'll have less people coming directly to you looking for those answers.

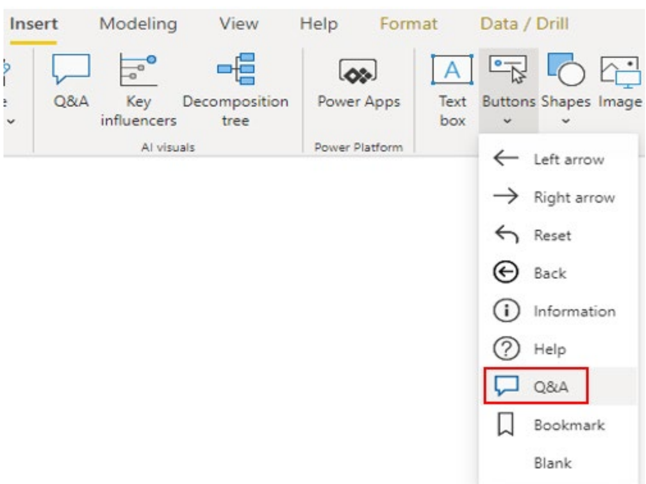
Suppose you've created a report for the Supply Chain team and the team members now have questions about various other views or insights they are interested in. You are getting inundated with these questions and don't have time to address each one individually. You decide to implement the **Q&A** feature, so that users can ask questions and get answers by themselves.

Add the Q&A visual to your report canvas

To get access to the **Q&A** feature, you need to add the **Q&A** visual to your report. You can simply double-click anywhere on the white canvas and the visual will appear. Alternatively, you can select the **Q&A** icon on the **Visualizations** pane.

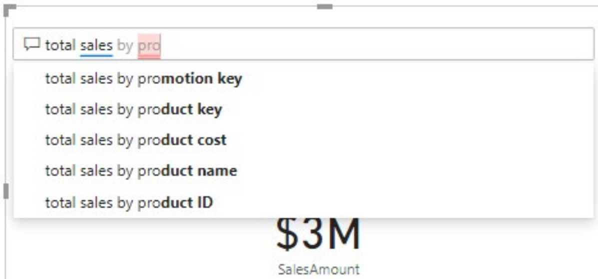


The **Q&A** feature is also available as a button, which is a useful option if you want to save space on your report canvas.



When the **Q&A** visual or button is added to your report, you can reposition and resize it, and customize its formatting, in the same way you would for any other type of visual or button.

You can start asking questions immediately by selecting one of the suggested questions or typing a question into the question box. As you type, Power BI will automatically display suggestions to help you complete your question.

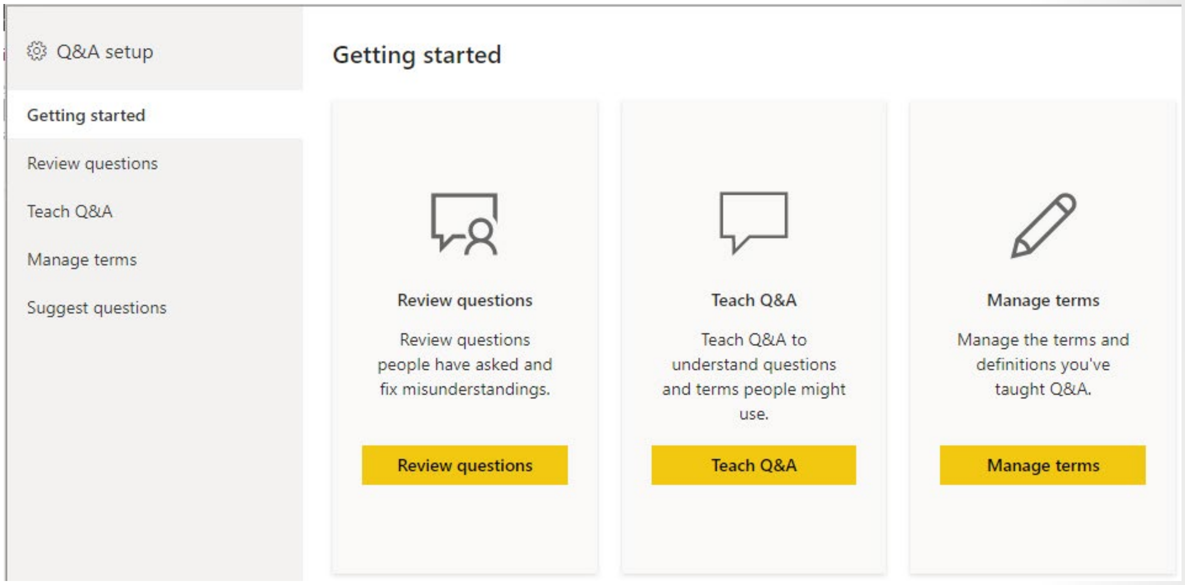


Set up the Q&A feature

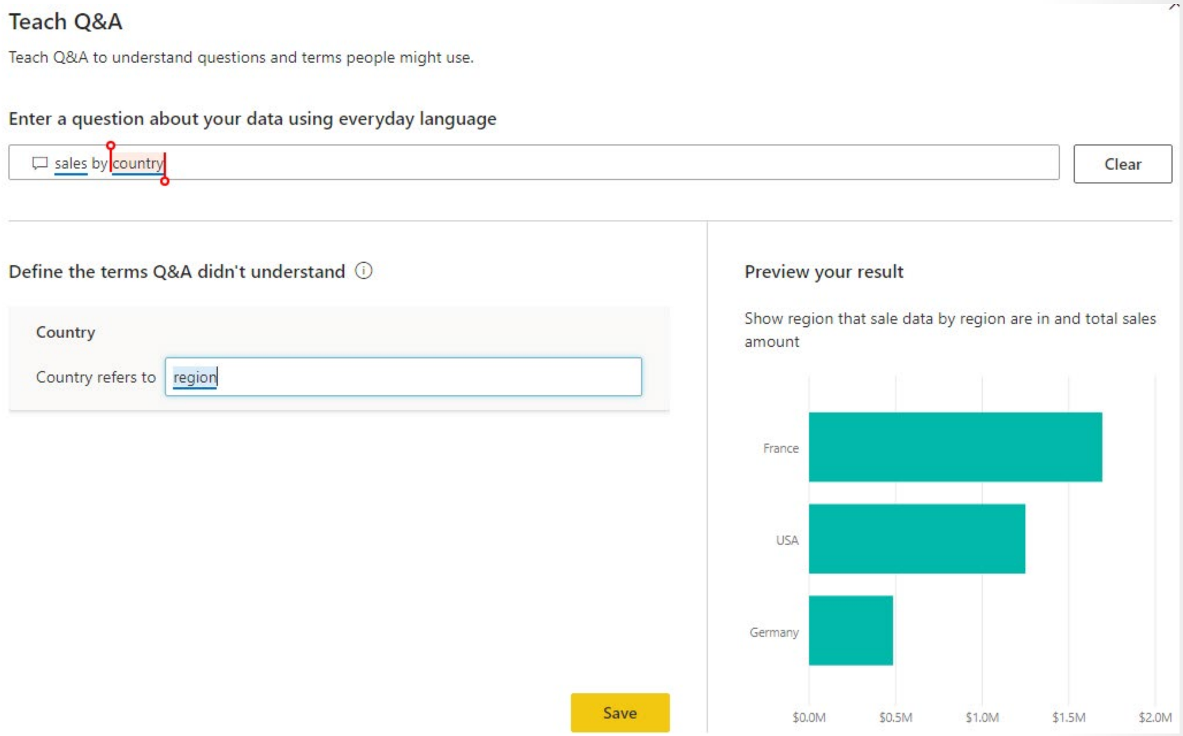
When you have added the **Q&A** visual to your report, you can set up the underlying Q&A feature, so that it becomes better at answering questions about your data. You basically teach the **Q&A** feature to understand people better. This set up can be useful from the outset, so you can get the visual ready for active use. However, set up does not stop there; you can proactively monitor and review the questions that are coming through from users, then and address misunderstandings or common typos. You can also manage the key terms associated with your data, so you can add a library of synonyms that may be used by different users across the organization when asking questions about the data. You can constantly fine-tune the **Q&A** feature, so it provides better answers to your organization's questions.

In this example, you get feedback from users saying that they can't get data on sales by country. You need to test this out, to see what the problem is, so you type the question "*sales by country*" into the question box. The visual does not update; it does not answer your question. As you can see in the following image, the word country is underlined in red. When there is a red underline, Power BI is telling you that it does not understand this term. You know that country is not used in your dataset, you use the term region instead. So that is why the question is not being answered. You need to teach Power BI what you mean by adding a new term to its thesaurus.

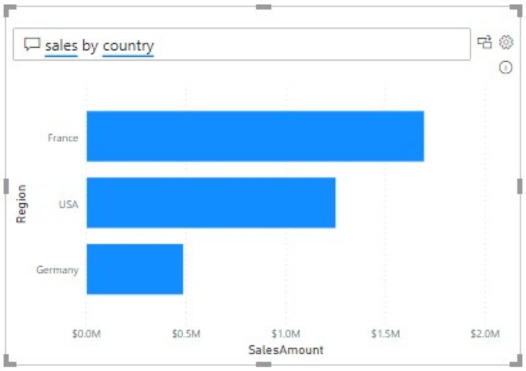
Select the settings icon to the right of the question box to open the **Q&A** setup window, then select the **Teach** Q&A option.



Enter your question again, then select the **Submit** button. In the **Define the terms Q&A didn't understand** section that displays, enter your alternative term, or synonym. In this case, you enter *region*. As you can see in the following image, Power BI displays a preview result, so you can see if this new term will return the results you are looking for. If this is correct, select **Save**.



Now when users search for sales by country, Power BI will know that they really mean sales by region, and automatically displays the associated data in the visual.

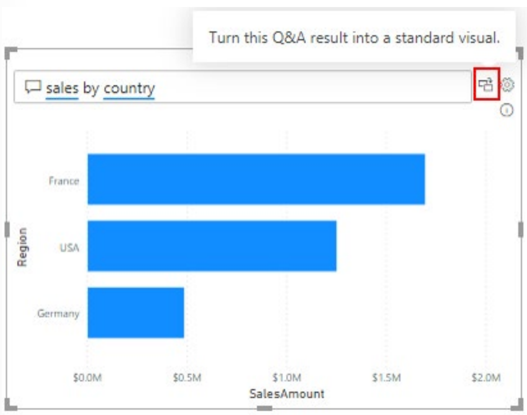


Build visuals using the Q&A feature

When Power BI answers a question and you find the visual result interesting or helpful, you can add it as a standard visual on your report.

For example, if you review the questions that are being asked and you see that a lot of users are asking the same question, you can make their life easier by adding the answer as a standard visual in the report, so they no longer have to ask the question. Similarly, you can also use the Q&A feature to start building your report, by asking questions and adopting Power BI's suggested visual result formats.

To turn a Q&A result into a standard visual, click the icon next to the question box.



The **Q&A** feature is unique in that it does not require knowledge of Power BI to use the visual, users simply have to ask their question and they too can create insightful visuals.

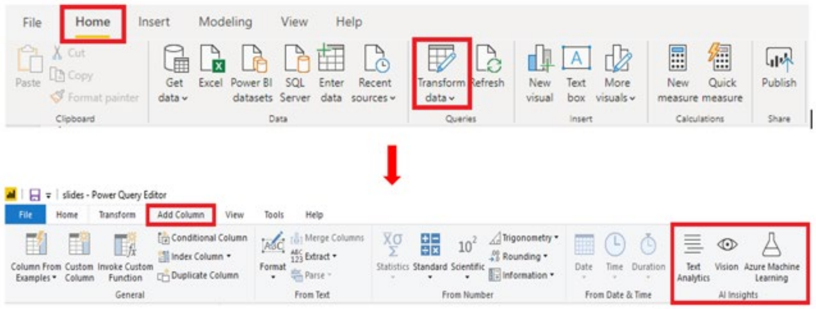
AI Insights

The **AI Insights** feature allows you to connect to a collection of pre-trained machine learning models that you can apply to your data, to enhance your data preparation efforts.

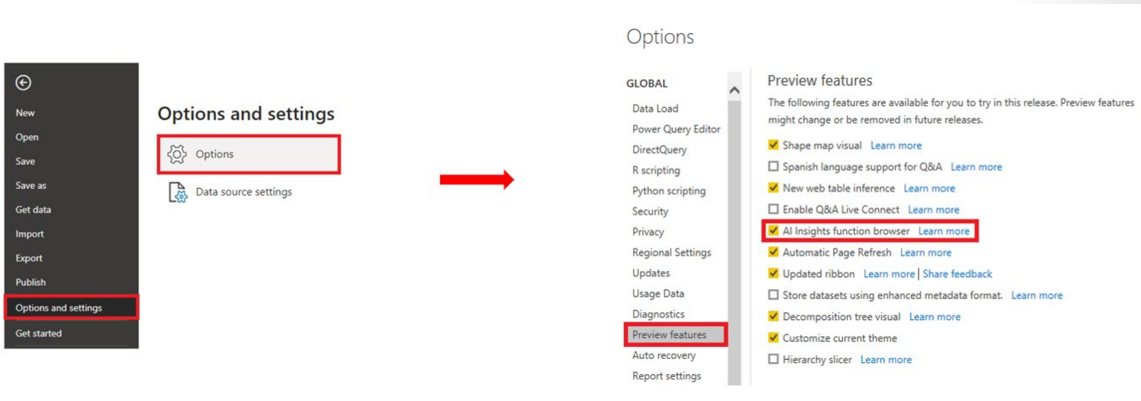
Continuing with the above example, suppose you now want to add text analytics to the content of the **Comments** field in the ticketing data, to see if you can determine the sentiment of the customers featured in the Help tickets. You can use the **AI Insights** feature to do that.

To apply the AI insights to your data, open Power Query Editor and select the **Add Column** tab. You'll see three **AI Insights** options: **Text Analytics**, **Vision** and **Azure Machine Learning**.

NOTE: Premium capacity is required to use the **Text Analytics** and **Vision** options.



NOTE: If you do not see these options, you need to enable the **AI Insights** feature in the Power BI Desktop settings. Go to **File** > **Options and Settings** > **Options**. In the **Global** options list, select **Preview Features**, then select the check box for the **AI Insights function browser** option, and then select **OK**.



On the **Add column** tab, the most relevant **AI Insights** option for this example is **Text Analytics**, which includes Azure Cognitive Services models, such as Sentiment Analysis, Key Phrase Extraction, and Language Detection that derive meaning or specific pieces of language from text data. You can use either the Sentiment Analysis or Key Phrase Extraction option to determine the customer sentiments in the Help tickets and visually show the results in Power BI.

Lseeon Review

In this lesson, you learned how to use the AI aspects of advanced analytics in Power BI to dig even deeper into your data and answer even more business questions.

You started by enhancing your reports with a self-service element; you added an interactive Q&A visual that allows users to get direct answers to the questions that they ask in their own words. You set up the Q&A feature to meet the needs of your organization and its dataset. You learned how you can continue to set up the Q&A feature going forward, as the data and organizational needs change, so that it works in the most optimized way and becomes a valuable asset in your reports. Next, you added the Key Influencers visual to your report, so users can understand the factors that are affecting a specific metric. Finally, you added a Decomposition Tree visual to your reports, to enable users to view your data across multiple dimensions and conduct root cause analysis.

The advanced analytical AI techniques you applied to your organization's data enabled you to take your data analysis to a whole new level. You gained deeper insights in the data by analyzing the 'why' and 'how' of the results of the data analysis. You were able to analyze a whole new set of data, data that was in text format and was untouched in your previous analysis.

The results of your AI advanced analysis will enable users at all levels of your organization to further analyze the data, get new insights, get answers to important questions, and address business challenges.

Knowledge Check

Question 2

Can you access the Q&A feature by using buttons?

- ☐ Yes, you can, but you will need to add the Q&A visual to your reporting canvas and then link your button with the visual that you have added.
- ☐ Yes, you can access the Q&A feature by selecting the Q&A button type.
- ☐ No, to use the Q&A feature, you will need to add the Q&A visual to your reporting canvas.

Question 3

What is not a feature of the Q&A feature?

- ☐ Searching for help topics about Power BI.
- ☐ Adding new synonyms to fields through Q&A tooling.
- ☐ Converting a Q&A answer into a visual inside your report.

Answers

Question 1

What Power BI feature can give an in-depth analysis of the distribution of data?

- ☐ The Next Level of Hierarchy feature can give in-depth analysis because it will allow you to drill down for all subcategories and is not used to analyze the distribution.
- ■ The Analyze feature allows a user to understand why the distribution of data looks the way it does.
- ☐ Only time series analysis can provide in-depth analysis on the data.

Question 3

What visual should be used to display outliers?

- ☐ The line chart is best-suited to display outliers.
- ☐ The clustered column chart is best-suited to display outliers.
- ■ The scatter chart displays outliers.

Question 2

Can you access the Q&A feature by using buttons?

- ☐ Yes, you can, but you will need to add the Q&A visual to your reporting canvas and then link your button with the visual that you have added.
- ■ Yes, you can access the Q&A feature by selecting the Q&A button type.
- ☐ No, to use the Q&A feature, you will need to add the Q&A visual to your reporting canvas.

Question 3

What is not a feature of the Q&A feature?

- ■ Searching for help topics about Power BI.
- ☐ Adding new synonyms to fields through Q&A tooling.
- ☐ Converting a Q&A answer into a visual inside your report.

Module 10 Create and Manage Workspaces

Creating Workspaces

Introduction to Workspaces

Through the past few discussions, you've had the chance to do a myriad of tasks, loading and transforming data from a number of sources, building visuals, creating DAX equations, maybe even publishing a report or two to Power BI Service, but what's next? The next step on our data analysis journey is to share these reports with our wider audiences and organizations. We can do this in a workspace, a feature of Power BI Service. A workspace is a centralized repository in which you can collaborate with colleagues and teams to create collections of reports and dashboards.

Workspaces offer the following benefits:

- Focus collaboration efforts as workspaces can be used to house reports and dashboards for use by multiple teams.
- Allow you to share and present reports and dashboards in a single environment.
- Ensure that the highest level of security is maintained by controlling who can access datasets, reports, and dashboards.

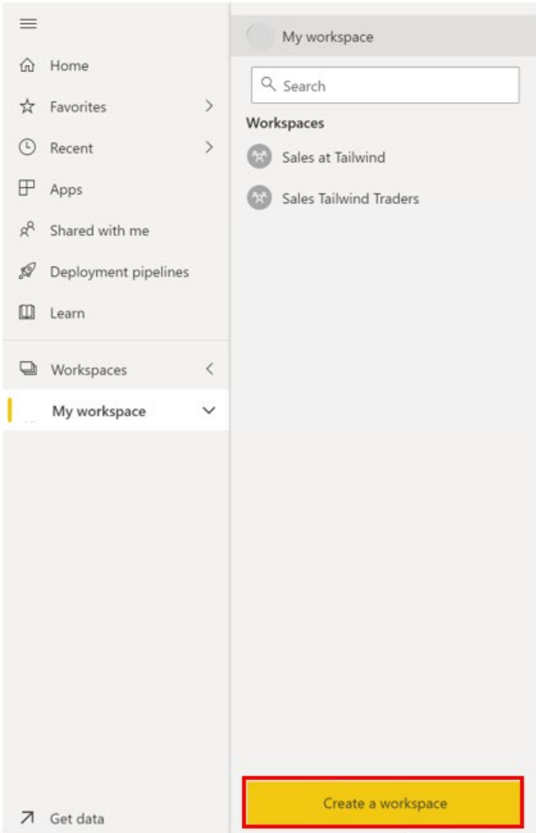
In the following discussions, we will speak on several tasks focused on creating and managing a workspace in Power BI. This includes importing and updating assets in a workspace, configuring data protection, troubleshooting data, and much more. With that, let's get started!

Create a Workspace

Suppose you have created a few reports for the Sales team at Tailwind Traders. How can you make these reports easily viewable and shareable? By creating a workspace in Power BI Service, you can house your reports in one location, make them shareable, collaborate with other teams, and easily make updates to reports. Let's see how we can do this!

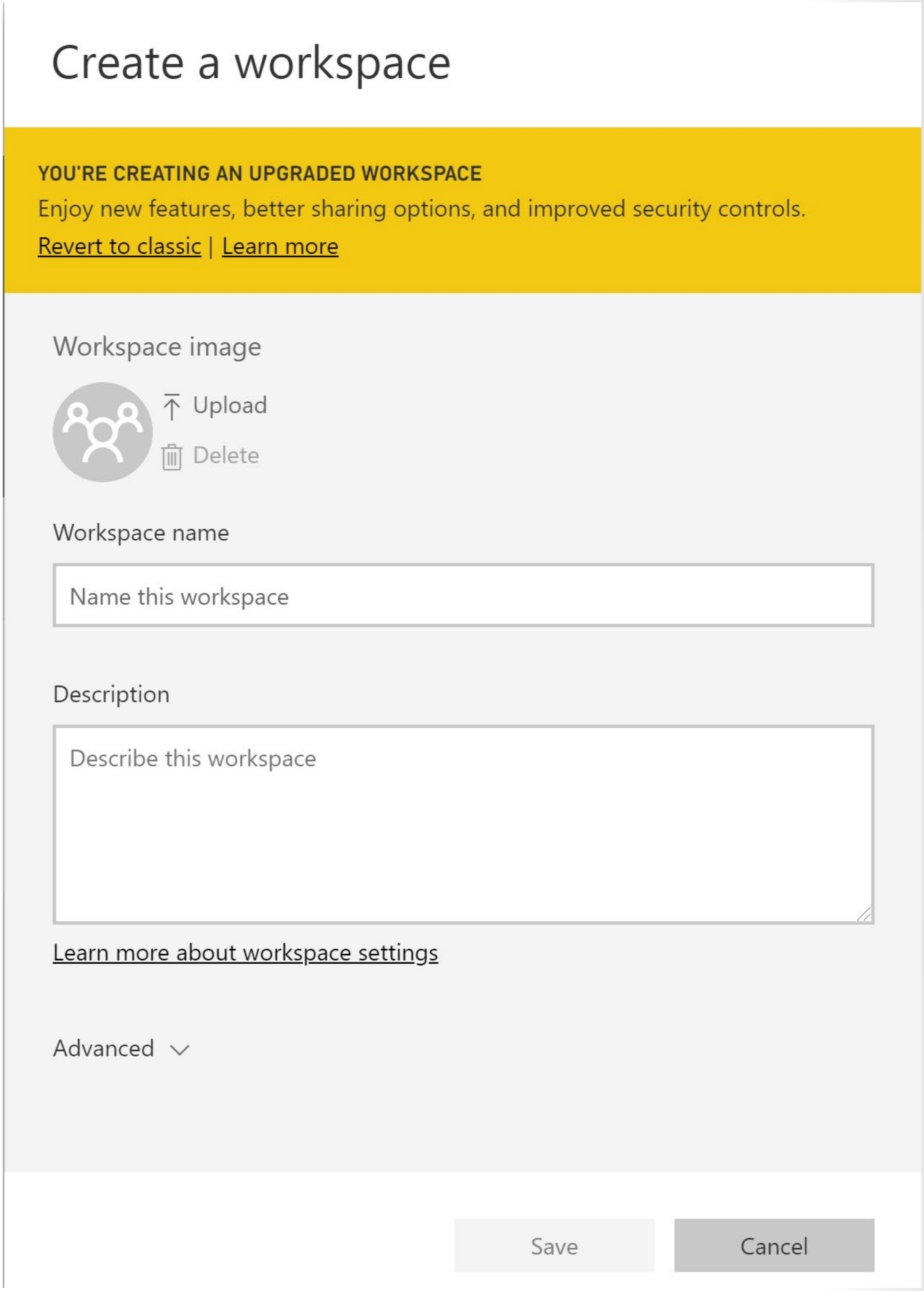
Create a workspace

Navigate to **Power BI Service**[1]. Select the dropdown on **Workspaces**. On the bottom of the resulting panel, you will see a **Create New Workspace** button.



Upon selecting **Create a Workspace**, you are brought to the following window, where you can add in a **Workspace name, Description,** and **Image**.

1 https://powerbi.microsoft.com



Under **Advanced**, you can create a **Contact list**, who is the users who will receive notifications if there are any issues with the workspace. By default, these are the workspace admins, but you can also add specific users. You can also add this workspace to a specific OneDrive and toggle to choose whether this workspace will be a part of a **dedicated capacity**. Dedicated capacities are a Power BI Premium feature that ensures that your workspaces will have its own computational resources as opposed to sharing resources with other users. Upon filling out the pertinent fields on this window, select **Save** and you've created a workspace! The previous discussion is done using the new workspace experience. As a recommendation, use the modern workspace experience over the classic workspace experience unless the classic workspace is expressly needed.

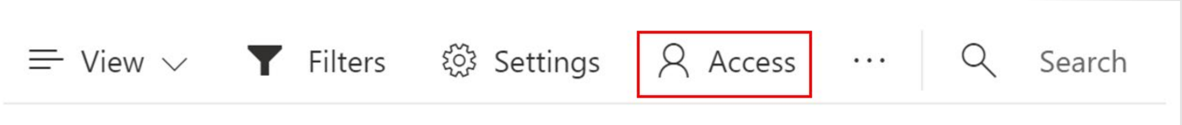
Assign workspace roles

You've successfully created a workspace and now Sales wants to collaborate with other team to build additional dashboards and reports. As the workspace owner, you want to ensure that the appropriate access is given to the members of the Products team since their team includes stakeholders and developers. Workspace roles allow you to designate who can do what within a workspace. The following are some of the abilities of role types in a workspace:

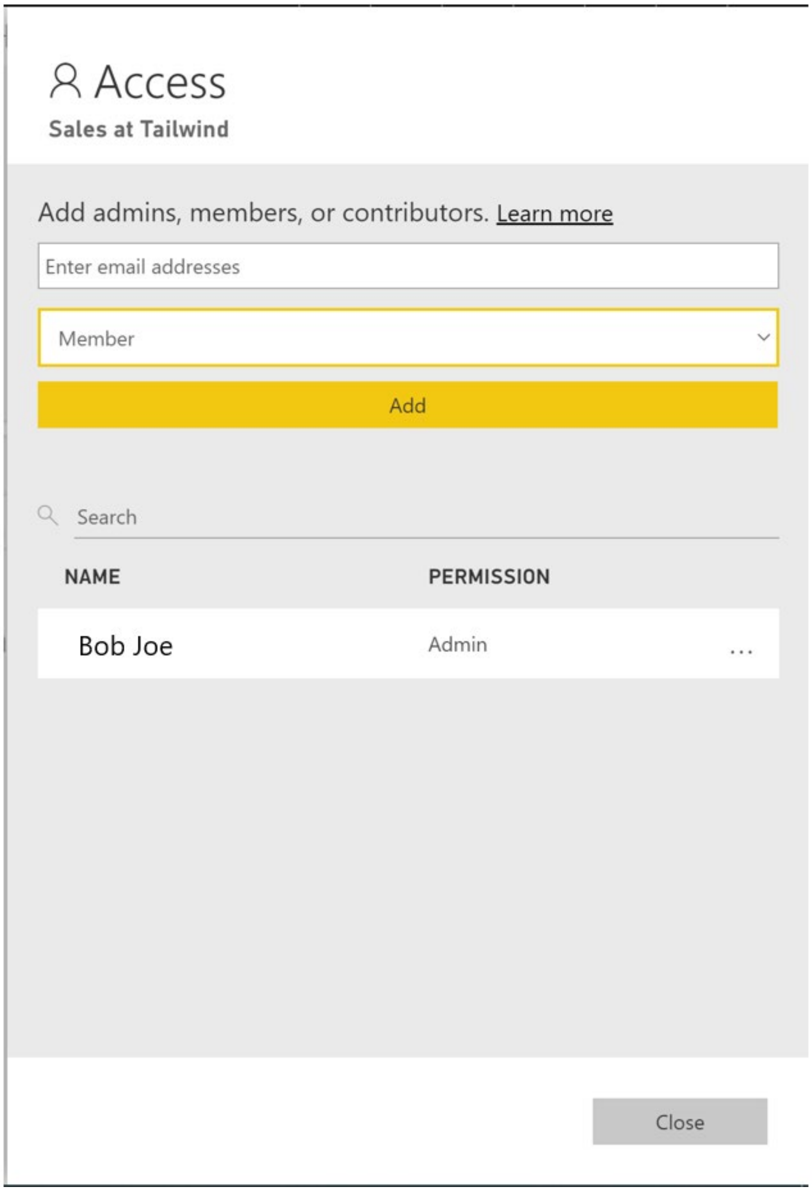
- Admin:
  - Add/remove other users
  - Publish, update, and/or share an app in a workspace
  - Create, edit, delete, and publish reports and content in a workspace
  - View and interact with reports and dashboards in a workspace
  - Configure data refreshes
- Member:
  - Can do all tasks associated with admins but **cannot** remove users
  - Can add users only as members or others with lower permission
  - Cannot delete the workspace
  - Cannot update the metadata about the workspace
- Contributor:
  - Cannot add or remove users
  - Cannot publish, update, or edit an app in a workspace unless given this ability by admins/members
  - Can create, update, and publish content and reports within a workspace
  - Can schedule data refreshes
- Viewer:
  - Cannot add or remove users
  - Can only view a report or dashboard in a workspace
  - Can read data stored in workspace dataflows

NOTE: If the workspace is backed by a Premium capacity, a non-Pro user can view content within the workspace under the viewer role.

To assign these roles to users, navigate to the workspace you've created and, on the top left of the ribbon, select **Access**.



In the resulting window, you can add email addresses of individual users, mail-enabled security groups, distribution lists, O365 Modern Groups, and regular security groups and assign them their specific roles. You can also change the user's assigned role at the bottom, as well as delete the user from the workspace here by selecting the ellipsis by their name.



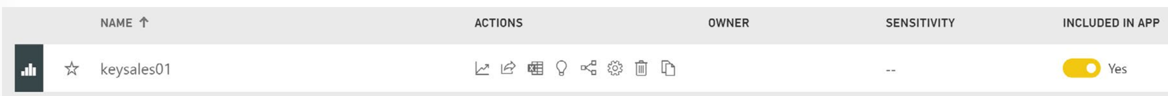
Create Apps

Now that you have created an app workspace and assigned your collaborators-specific roles, you want to be able to add content to your app workspace. Content can be in the form of reports, dashboards, datasets, dataflows, etc. An app is a published read-only window into your data for mass distribution and viewing. When you are ready to share apps with your users, you can **publish the app**. This process requires a Power BI Pro license. Consuming and viewing an app requires a pro license or backed premium capacity.

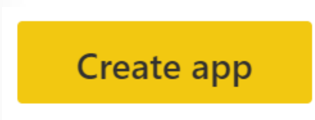
After you have added your content to the App workspace, then you need to create the app. Navigate to your workspace, and on the ribbon, select **+ New**. Here, you can choose to create a new report, dataset, streaming dataset, or dataflow to name a few. Selecting any one of these options will bring up a window where you can add in the name of the app and select the source of the report (for example, the dataset used to create a report).

You can also select the **Get Data** button on the bottom-left of the navigation bar and import already-existing reports from Power BI Desktop and add them to your workspace app.

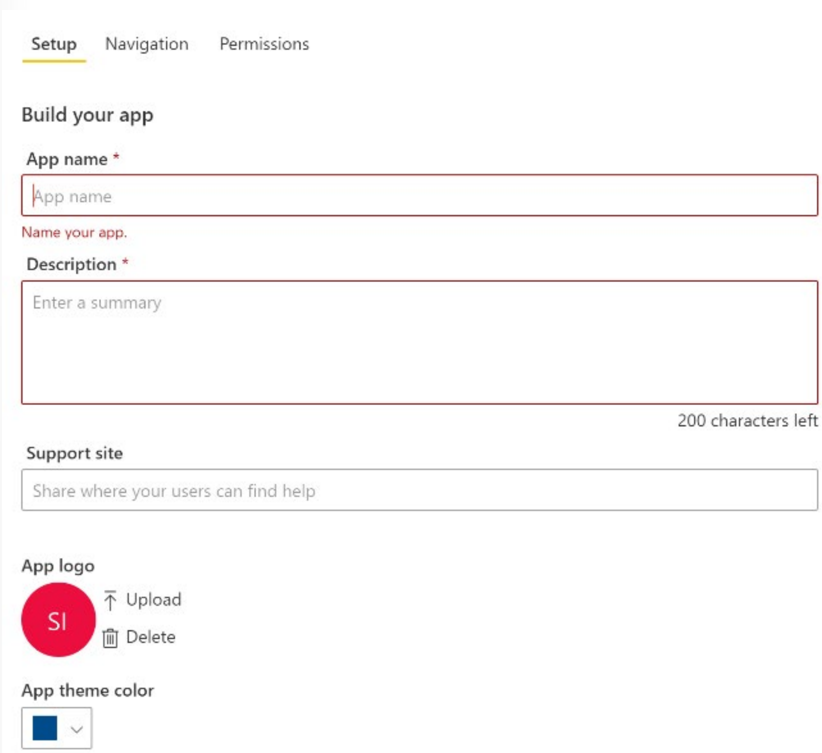
You can also configure your app and enable whether or not you'd like to include the report or dashboard in the app when you publish, as seen in the following. If you do not want to include this report or dashboard in this app, then you can toggle off this option.



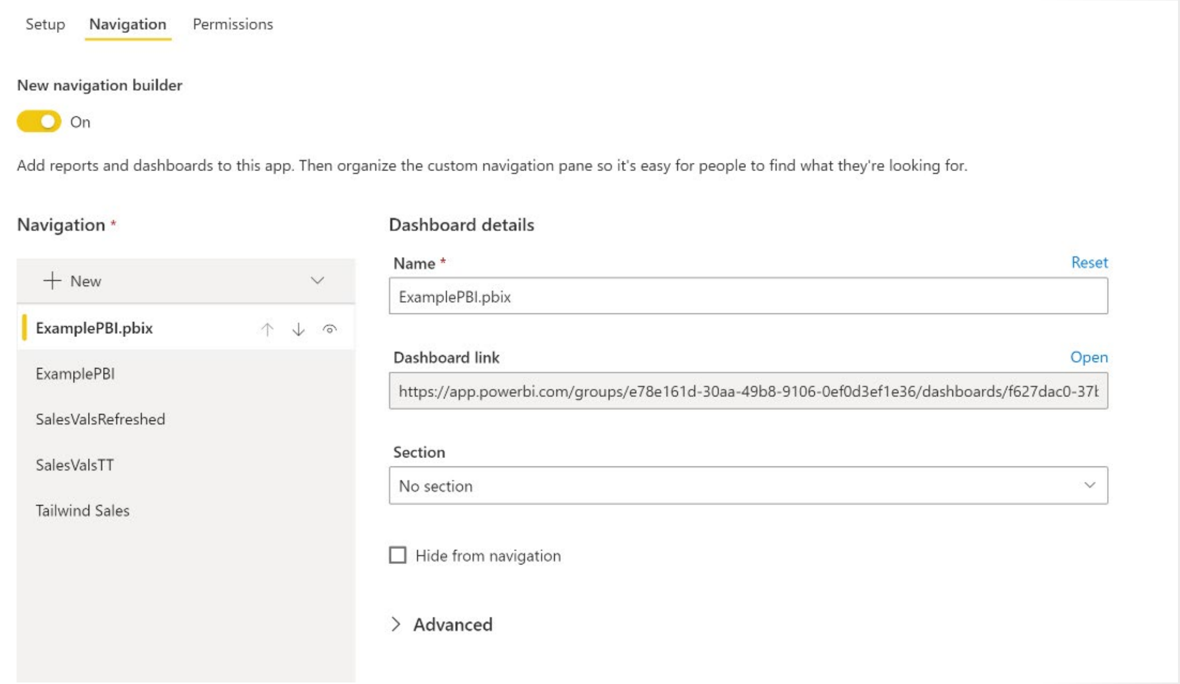
When you are ready to publish your app with its collection of reports, dashboards, and datasets, navigate back to the workspace, and select **Create App** in the top-right corner of the ribbon.



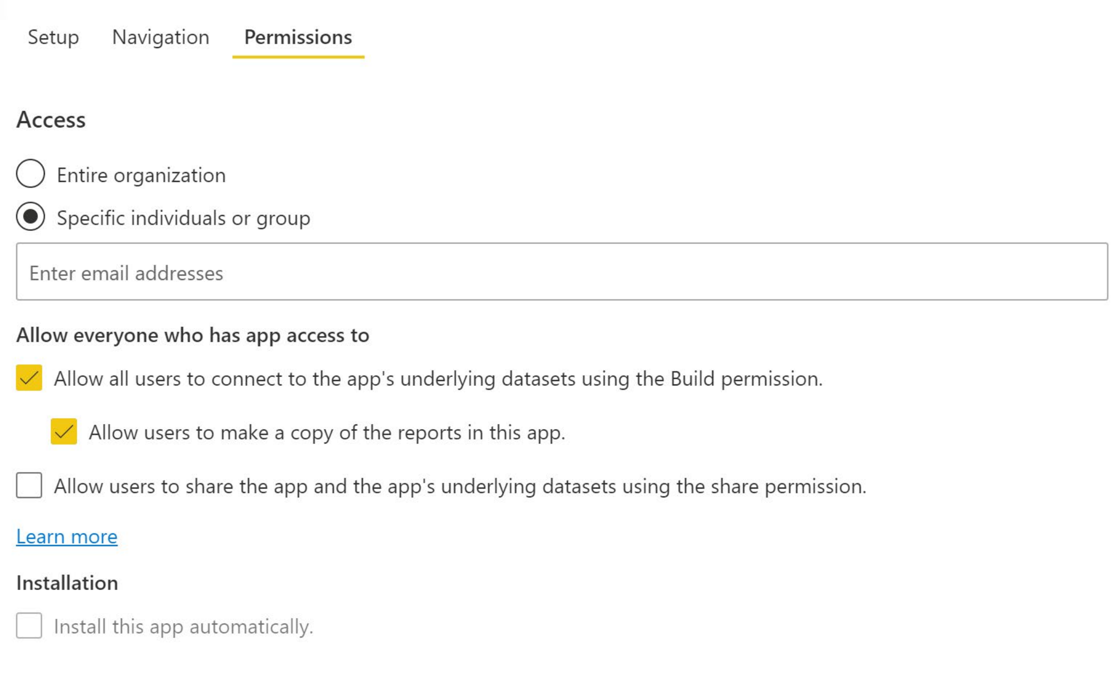
This retrieves the following window, where you can build your app by naming it, adding a description, and adding an app logo or theme color, if needed.



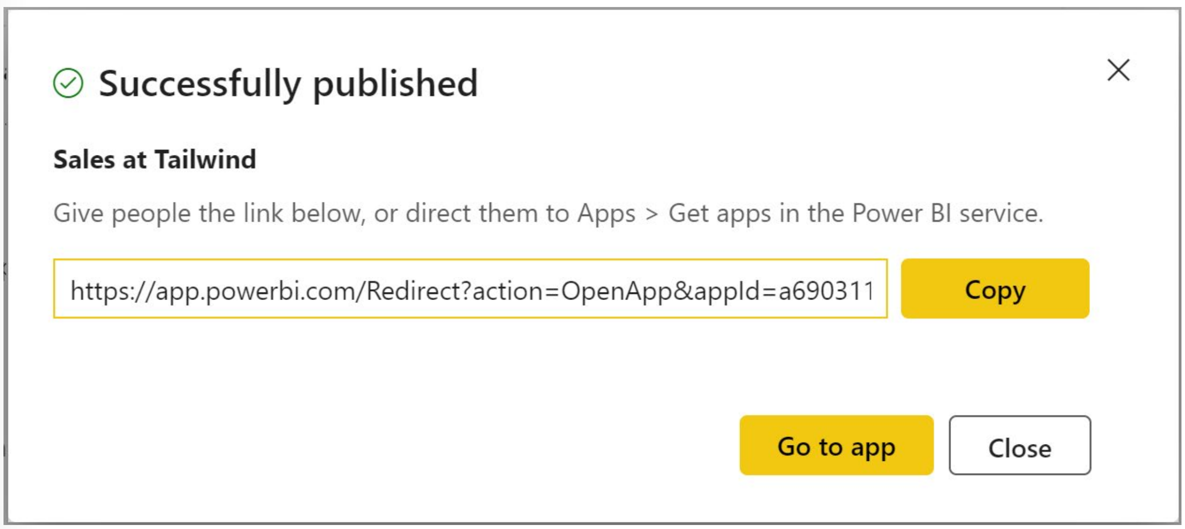
Under the **Navigation** tab, you can change the order in which the content is oriented for the user by creating a custom navigation pane. For instance, you can change the name of the content, change the link, and add it to a specific section on the navigation pane. You can also add content external to Power BI via a link. This can also be included within the content area. For example, a YouTube video, or PowerPoint slide deck has to be an embed URL though, not the raw URL



Under the **Permissions** tab, you can grant access either to all the users in your organization or you can choose which users have access. You can also give your users build and share permissions, which mean that they can create and share the content in the app. There are a few options here: with Build permissions, you allow your users to connect to underlying datasets so that they can reuse and build out their own reports using the same dataset. Build permissions are required if your users would like to export the underlying data or to build new content on top of the data. You can also allow your users to only create a copy of the report to view in another workspace, where they can modify and delete visuals as per their needs. You can also give your users Share permissions so that they can share underlying datasets and reports.



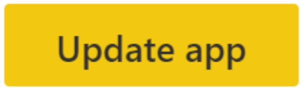
Upon making edits were necessary, you can select **Publish App** and voila you've published an app! You will get the following screen with a link that you can share with your users:



Configure and update a Workspace App

After publishing your app, you realize that you would like to make updates within your workspace. How can you do this?

Navigate back to the workspace, and make any updates required in the reports or dashboards. The workspace acts as a staging area you can make any changes but they will not be added to the app until you select **Update App** in the top-right corner of the ribbon (where previously we had **Publish App**). Dataset and dataflow updates are updated immediately. When you select this, you can make any changes to the **Setup, Navigation,** and **Permissions** tab, and when ready, select the **Update App** button.



If you are interested in learning more, please refer to **Publish an App in Power BI**[2].

Knowledge Check

Question 1

How does the admin workspace role differ from other types of workspace roles?

☐ Admin is the only role that can publish or update apps.

☐ Admins are the only role that can remove users.

☐ Admin is the only role that can create, edit, or delete content in a workspace.

☐ Admin is the only role that can publish content to a workspace.

Question 2

What is the best description of a workspace?

☐ A workspace is a feature in Power BI service that allows you to view reports only.

☐ A workspace is a feature that allows you to view and edit the data model, build visualizations, and transform the data.

☐ A workspace is a feature of Power BI Desktop that allows you to build reports only.

☐ A workspace is a centralized location or repository that allows you to collaborate with colleagues and teams to create collections of reports, dashboards, etc.

2 https://docs.microsoft.com/power-bi/collaborate-share/service-create-distribute-apps

Sharing and Managing Assets

Monitor usage and performance

You've successfully added reports to your workspace, published an app, and begun the process of collaborating with the Products team. News of how useful workspaces are begins to spread around Tailwind and more users get added to the workspace. The Sales team knows that performance may reduce as more users get added so would like you to monitor usage and performance of the workspace.

Knowing how your workspace is being used and is performing is crucial for several reasons:

- Focuses your efforts for improvement; if you know where you have the least performance you can concentrate your efforts for improvement in those areas.
- Quantifies the impact of your reports; by seeing usage metrics you can determine how successful your reports are.

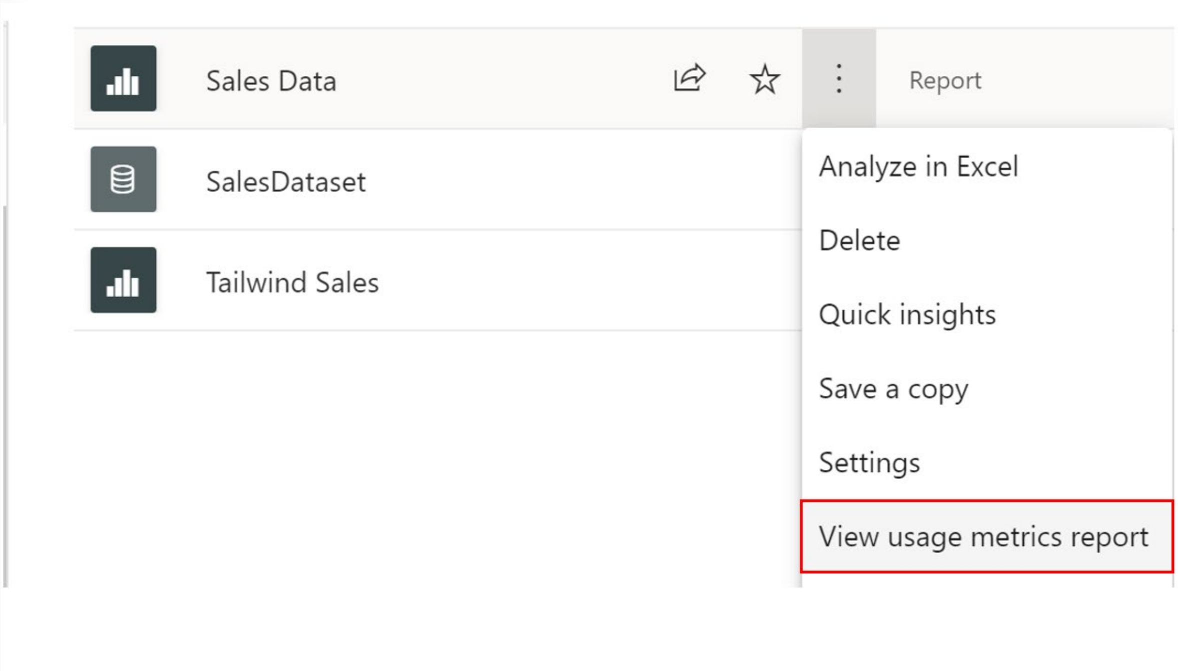
These performance and usage metrics are both a feature available for use in a workspace. You can see who's using your reports, what actions are being done on the reports, and what performance issues exist.

Let's take a closer look!

Configure and view usage metric reports

Usage metric reports are available for Power BI pro users and can only be accessed by users with role types admin, member, or contributor.

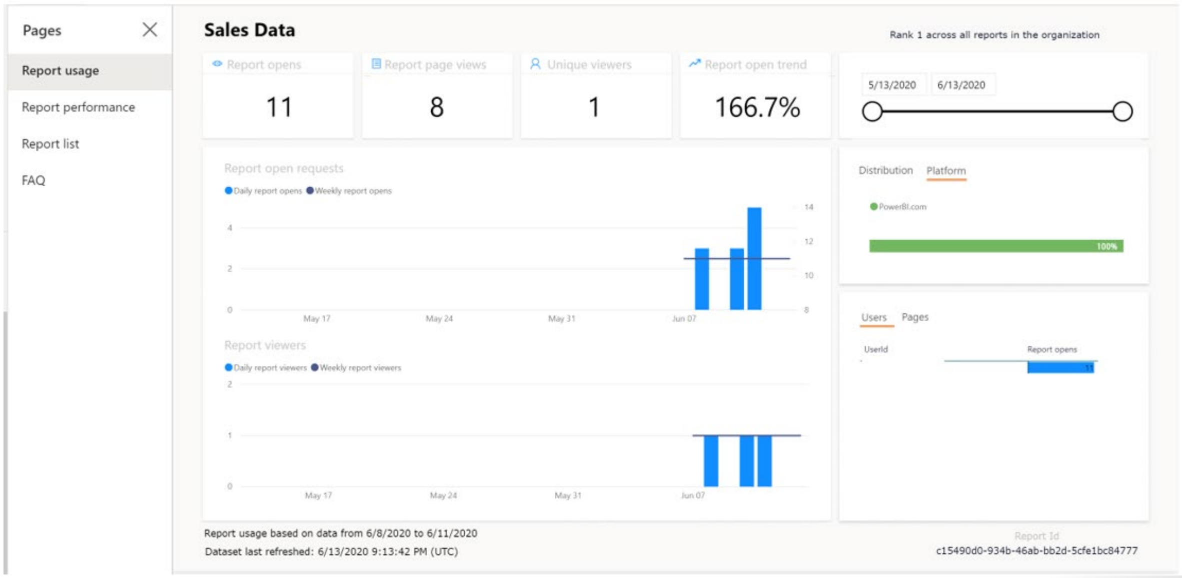
Navigate to the pertinent workspace. Then, find the report or dashboard you would like to see usage metrics for. Let's say we want to see the usage metrics report for **Sales Data**. Under the ellipsis, select **View usage metrics report**.



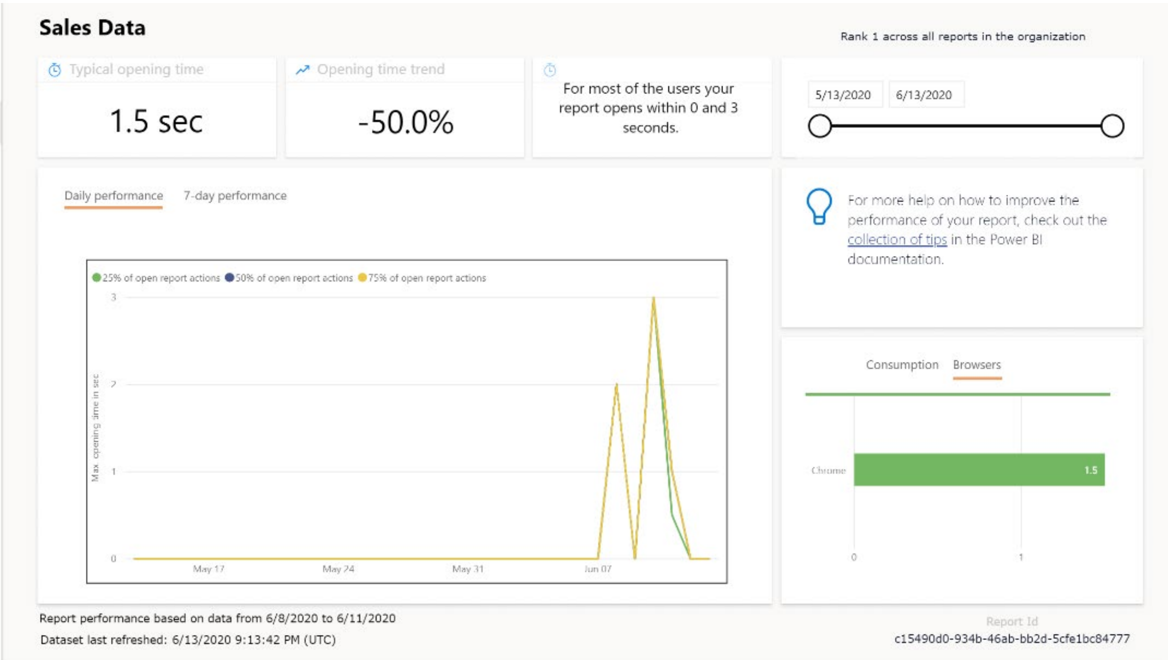
Then you will receive a prompt when the usage metrics report is ready for viewing. When you do, you be brought to a dashboard in which the first tab, **report usage**, you can view such details as:

- **Viewers per day**, **Unique viewers per day** (which doesn't include users who come back to the same reports multiple times), **Shares per day charts**
- **Total views**, **Total Viewers**, and **Total Shares** KPI cards
- **Total views and shares Ranking**: Compares how your report is doing in comparison to other reports in the app
- **Views by Users**: detail about each specific user that viewed the dashboard.

You can also filter by the distribution method of the report (for example, through sharing or from the workspace directly) and platform type (for example, mobile or web).



You can also view performance metrics on the next tab, **Report performance,** as seen in the following.



Here, you can see metrics such as:

- **Typical opening time**: how long it takes, at the 50th percentile, to open the report.
- **Opening time trend**: how the typical opening time changes over time. This can tell you how the report is performing as the number of users starts to grow.
- **Daily/7-Day Performance** charts: highlights the performance for 10%, 50, and 90% of the open-report actions every day and over a seven-day period.
- Filters for date, so you can see how the performance changes according to the day.

With that, you learned about the basics of monitoring usage and performance. If you would like to know more, please refer to **Monitor Usage Metrics**[3].

Viewing data lineage

You've had the chance to develop many reports and have published them to the Tailwind workspace. However, because you are also collaborating with the Products team, it has become increasingly difficult to track which reports need to be refreshed and even which datasets are in which report, it is a jumbled mess! You want to be able to determine which datasets need to be refreshed as you've been getting reports of stale data. The path of data from its source to the destination can often be a gargantuan challenge, and more so if you have multiple datasets to add to the mix. How can you make sense of the intricacies of the data paths and troubleshoot unrefreshed data?

Power BI's Lineage View allows you to do just that. In Lineage View, you can quickly refresh datasets and see the relationships between the artifacts in a workspace as well as their external dependencies. With that, let's get started!

3 https://docs.microsoft.com/power-bi/collaborate-share/service-modern-usage-metrics

Data lineage

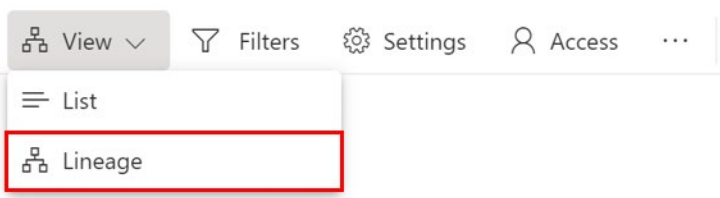
Data lineage refers to the path that data takes from the data source to the destination.

View lineage in Power BI is crucial for several reasons:

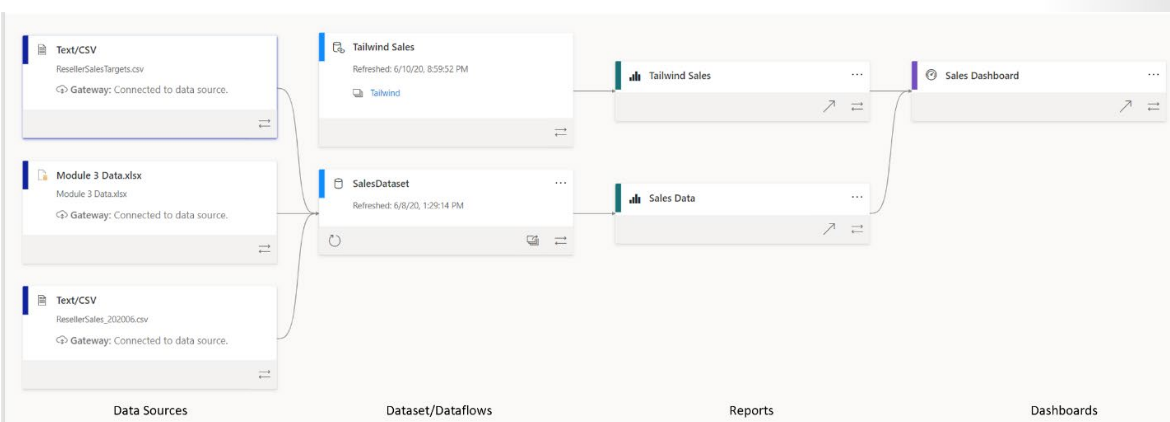
- Simplifies the troubleshooting process, as you can see the path the data takes from source to destination and easily determine pain points and bottlenecks.
- Allows you to manage your workspaces more easily and clearly see the impact of a single change in one dataset to reports and dashboards.
- Saves time by making it easy to identify reports and dashboards that haven't been refreshed.

Using lineage view

Lineage view is only accessible to admin, contributor, and member roles. Additionally, lineage view requires a Power BI Pro license and is only available for App workspaces. First, navigate to the workspace. Select **Lineage** from the **View** dropdown menu on the top ribbon.

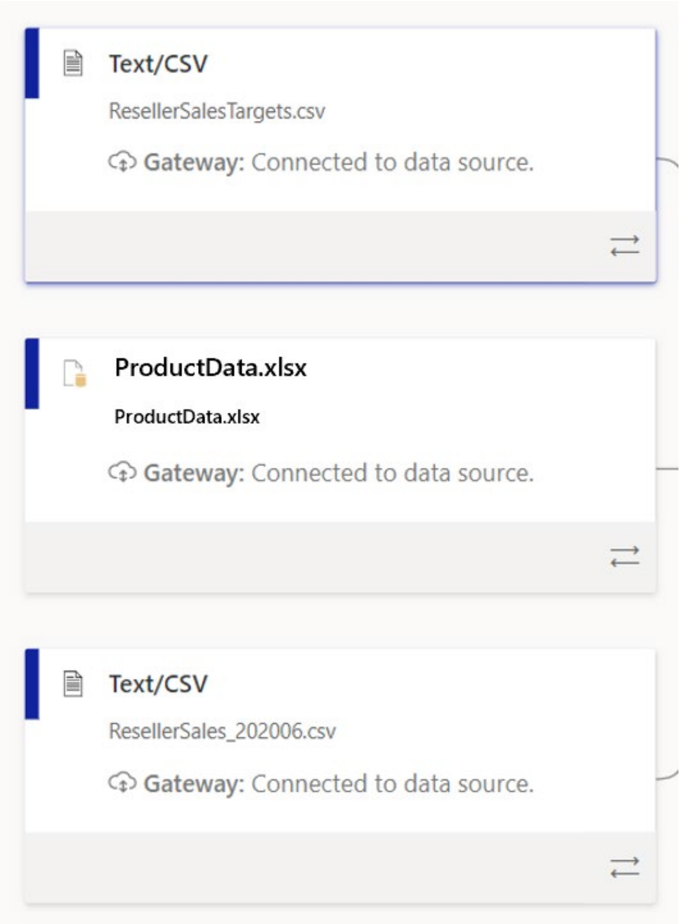


Once the view canvas opens, we can begin to explore this view. The following is an excerpt of the data lineage for our workspace, **Tailwind Sales**.



This view shows us all the artifacts in our workspace. Artifacts include data sources, datasets and dataflows, reports, and dashboards. Each card represents an artifact, and the arrows in between these cards represent the flow of data, or the relationship between different artifacts. By following the arrows from left to right, we can see the flow of data from the source to the destination, which will often be a dashboard. Our flow is typically the following: data sources > datasets/dataflows > reports > dashboards. Let's look at these steps.

Data sources

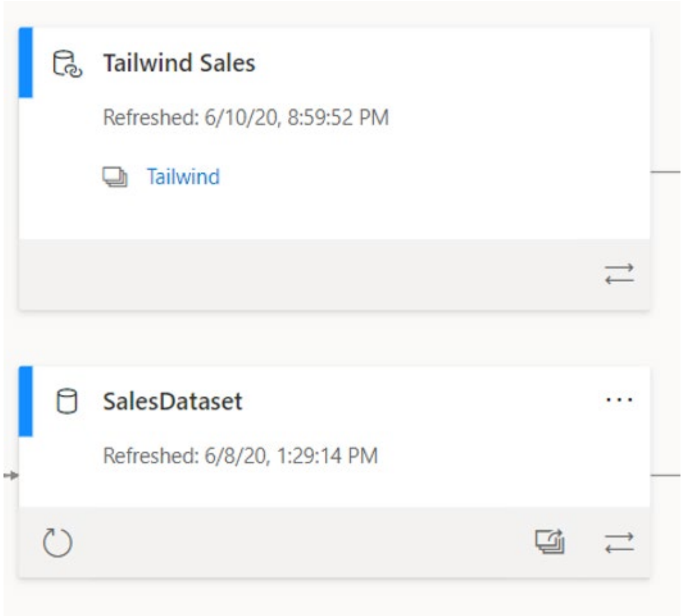


Each of the above cards is a data source used in our workspace. This card tells you the type of data source (for example, **Text/CSV**) and the gateway. **Gateway** tells us the source of our data. If we are connected to the data via an on-premises data gateway, this will tell you more information about the gateway. Additionally, double-clicking the card itself will you tell more details about the data source, such as the file path and the connection status. Selecting the bottom-right icon on the card will highlight specifically the path from the data source to the destination, which makes it clearer as to the exact path the data takes.

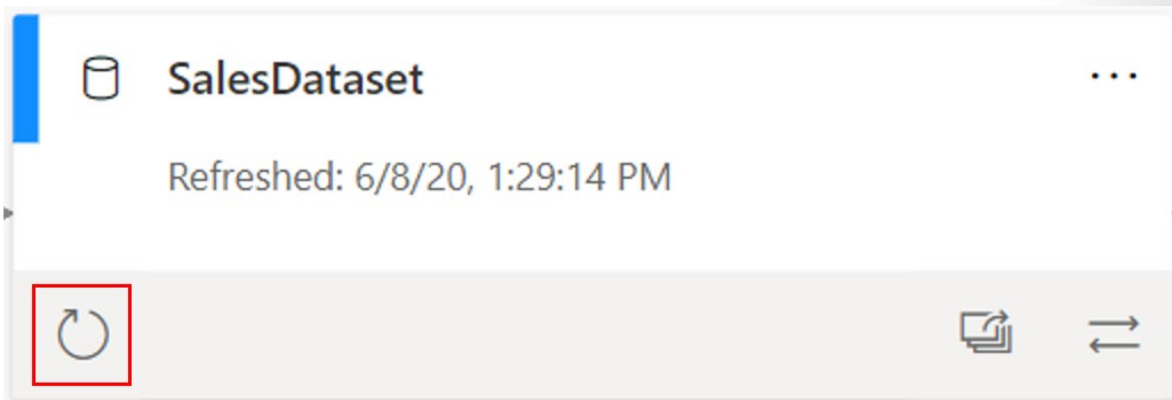
Next, we have datasets/dataflows, marked in blue.

Datasets/dataflows

Often datasets and dataflows can connect to external data sources, such as SQL Server or to external datasets in other workspaces. The following are examples of dataset and dataflow cards on the lineage view.

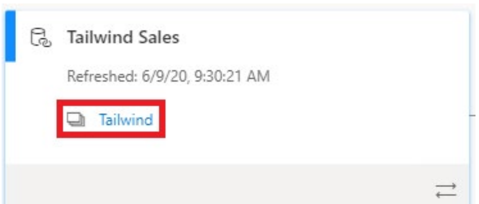


The lineage view uses arrows to connect objects, such as datasets, with their data sources. On these cards you can see when the dataset was last refreshed, as well refresh the dataset itself by selecting the arrow icon on the bottom left of the card, as seen in the following.

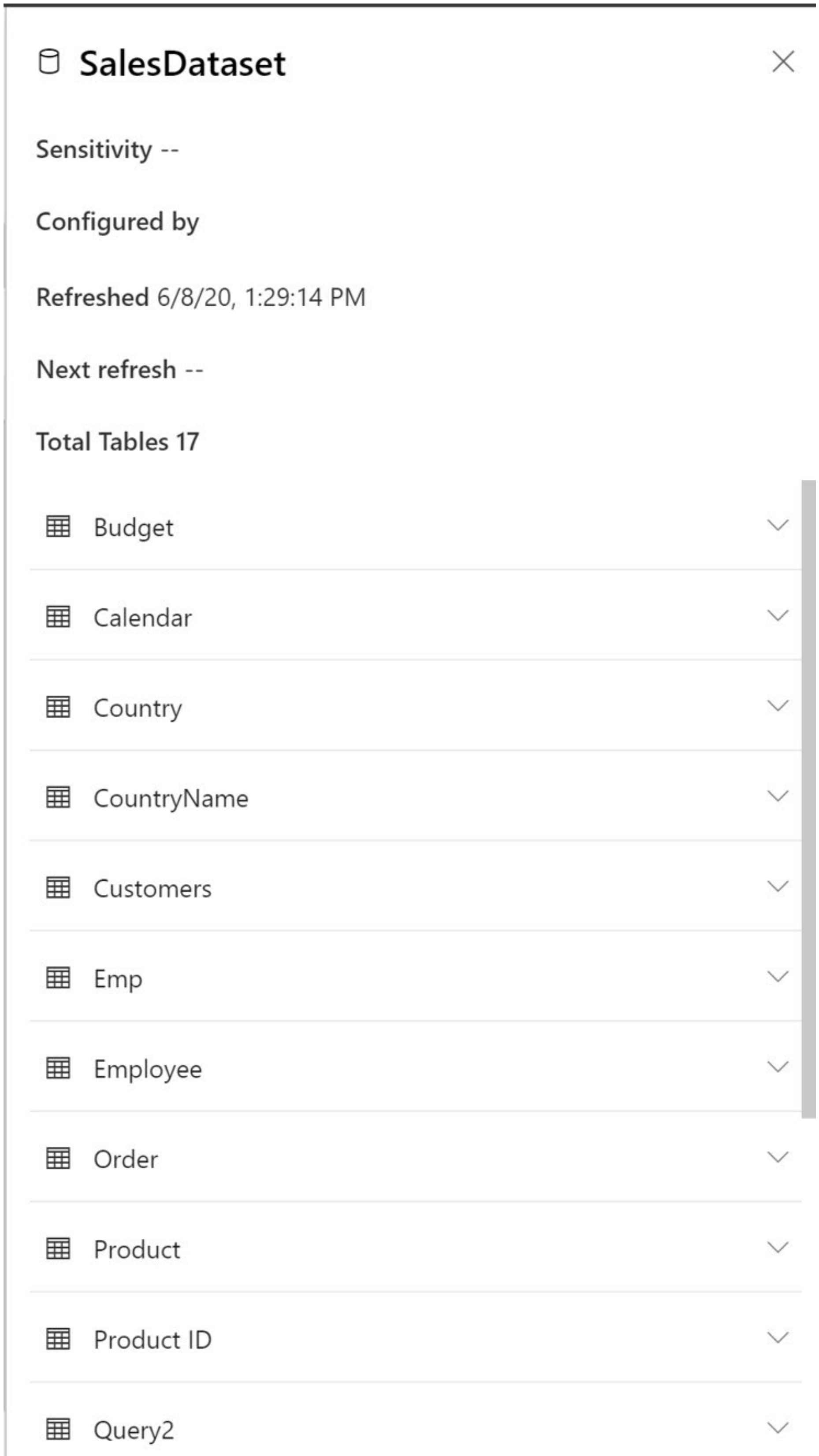


This is a powerful troubleshooting feature that makes it a quick and painless task to ensure that all your datasets are refreshed. Going back to our initial quandary, we wanted to easily see if we have any stale datasets and then quickly be able to refresh the data. Using this, you can go through the different datasets in one view and use this button to refresh any datasets that you believe are stale.

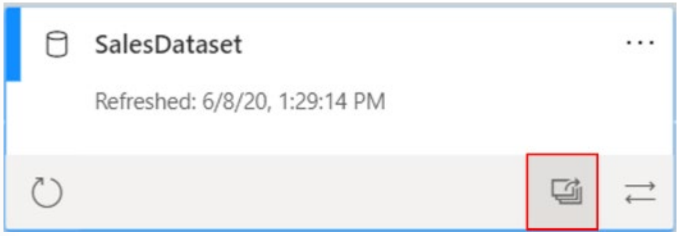
Additionally, if a dataset or dataflow belongs to a different workspace (in this case, the **Tailwind** workspace), it will be indicated on the card, as seen in the following.



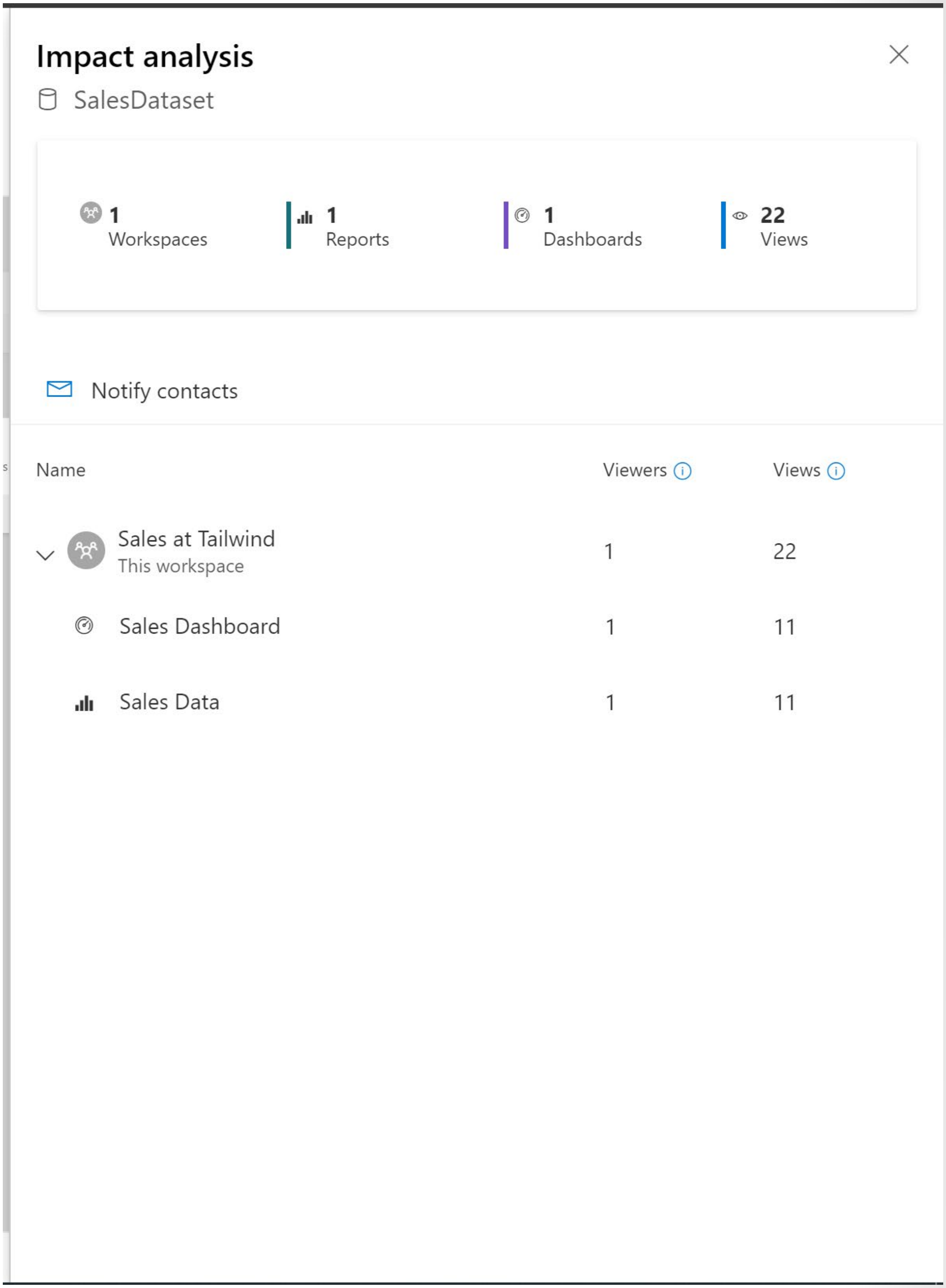
There are a few other features that are important to mention. First, by double-clicking on any card, you can view the metadata. You can see the sensitivity, whom it was configured by, the last refresh date, and the names and count of tables within this dataset.


SalesDataset
Sensitivity --
Configured by
Refreshed 6/8/20, 1:29:14 PM
Next refresh --
Total Tables 17
Budget
Calendar
Country
CountryName
Customers
Emp
Employee
Order
Product
Product ID
Query2

You can also view the impact of this dataset across workspaces. By selecting the overlapping window icon on the bottom right of a dataset card, as seen in the following, you can determine the impact analysis of the dataset.



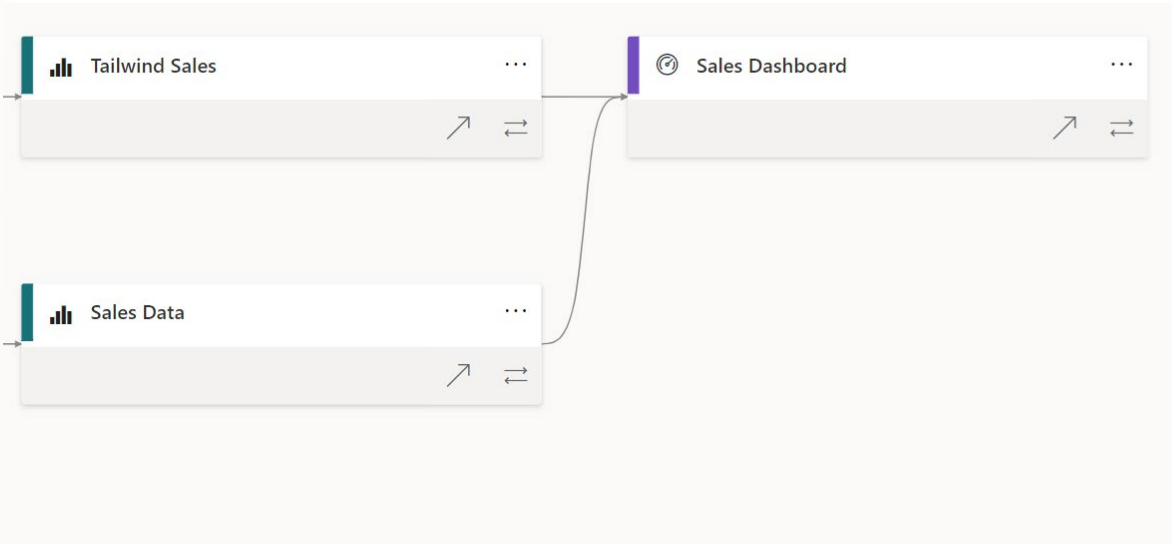
On the **Impact analysis** window, you can see how many workspaces, reports, and dashboards this dataset is a part of, as well as how many views this dataset has gathered, as seen in the following.



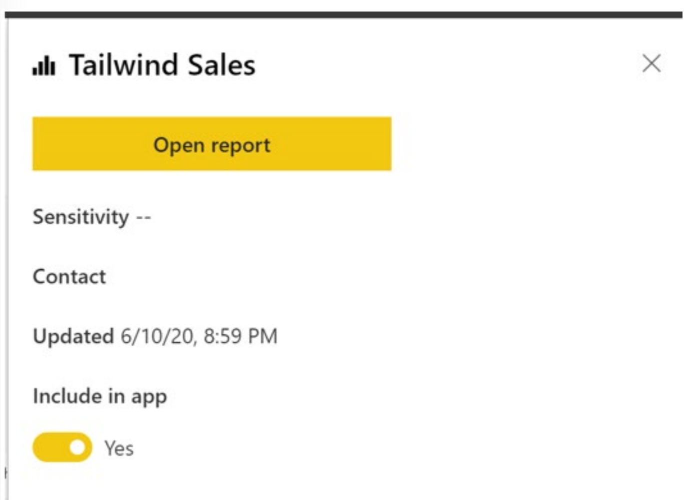
At the bottom of this window, you can also see more detail about which specific reports and dashboards this dataset is a part of. Additionally, you can **notify contacts**, which allows you to notify dataset owners (or any other user) of changes in the dataset. Impact analysis is useful for it allows you to pinpoint if you have any datasets that aren't being used or looked at.

Reports/Dashboards

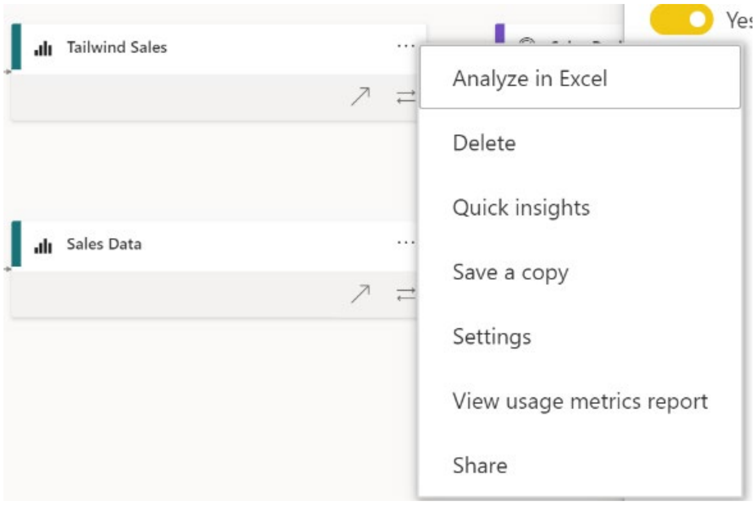
The reports and dashboards cards have similar functionality as the data source and dataset cards do.



Selecting a card will bring up a window in which you can view the metadata about the report or dashboard; here, you can also go directly to the report or dashboard. You can also enable or disable whether you would like to include this report or dashboard within the app.



This card also contains useful options under the ellipsis, as seen in the following. Here, you can analyze the report in Excel, delete a report, create Quick Insights, save a copy directly onto your local drive, and more.



With that, you have had a chance to take an in-depth look into the Lineage View in Power BI Service. Knowing this, you can go forth with confidence and tackle cleaning up the Tailwind workspace. If you are interested in learning more, please refer to **Data Lineage**[4].

Data Protection

As enterprises grow, so do their data. There are often strict requirements and regulations that must be applied to ensure that this sensitive data is secure. There are a few ways you can do this in Power BI:

- Use Microsoft Information Protection sensitivity labels to classify critical content in Power BI without compromising productivity or the ability to collaborate.
- Add additional protection measures such as encryption and watermarks when exporting the data.
- Use Microsoft Cloud App Security to monitor and investigate activities in Power BI.

As more and more reports and dashboards are being added to the Tailwind workspace, concern grows as the Sales team realizes the urgency of securing its data. There's a worry about the possibility of new users exporting data without permission. Sales doesn't want to roll back any reports or dashboards, and have come to you to implement comprehensive security measures that protect data access within and outside of Power BI. This can be done by configuring Microsoft Information Protection sensitivity labels in Power BI. Before we begin, ensure that you have the appropriate licensing and requirements, as seen **here**[5].

Microsoft Information Protection sensitivity labels

Microsoft Information Protection sensitivity labels provide a simple way to classify critical content in Power BI without compromising productivity or the ability to collaborate. They can be applied in both Power BI Desktop (preview) and the Power BI service, making it possible to protect sensitive data from the moment you first start developing your content on through to when it's being accessed from Excel via a live connection. Sensitivity labels are retained when you move your content back and forth between Desktop and the service in the form of .pbix files.

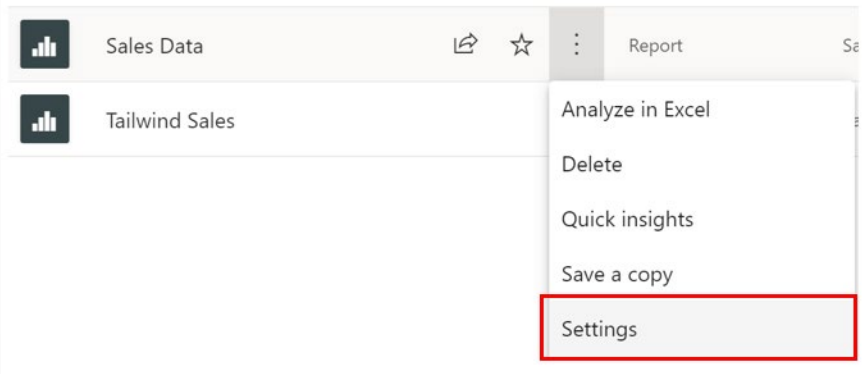
In the Power BI service, sensitivity labels can be applied to datasets, reports, dashboards, and dataflows. When labeled data leaves Power BI, either via export to Excel, PowerPoint, PDF, or .pbix files, or via other supported export scenarios such as Analyze in Excel or live connection PivotTables in Excel, Power BI

4 https://docs.microsoft.com/power-bi/collaborate-share/service-data-lineage
5 https://docs.microsoft.com/en-us/power-bi/admin/service-security-enable-data-sensitivity-labels

automatically applies the label to the exported file and protects it according to the label's file encryption settings.

In order for Microsoft Information Protection sensitivity labels to be used in Power BI, they must be enabled on the tenant. When data protection is enabled on your tenant, sensitivity labels appear in the sensitivity column in the list view of dashboards, reports, datasets, and dataflows.

Once you have verified your ability to add labels, navigate to any workspace, and choose an object to secure. In this case, I want to add a sensitivity label to **Sales Data**. To do this, navigate to the workspace, and under the ellipsis, select **Settings**.

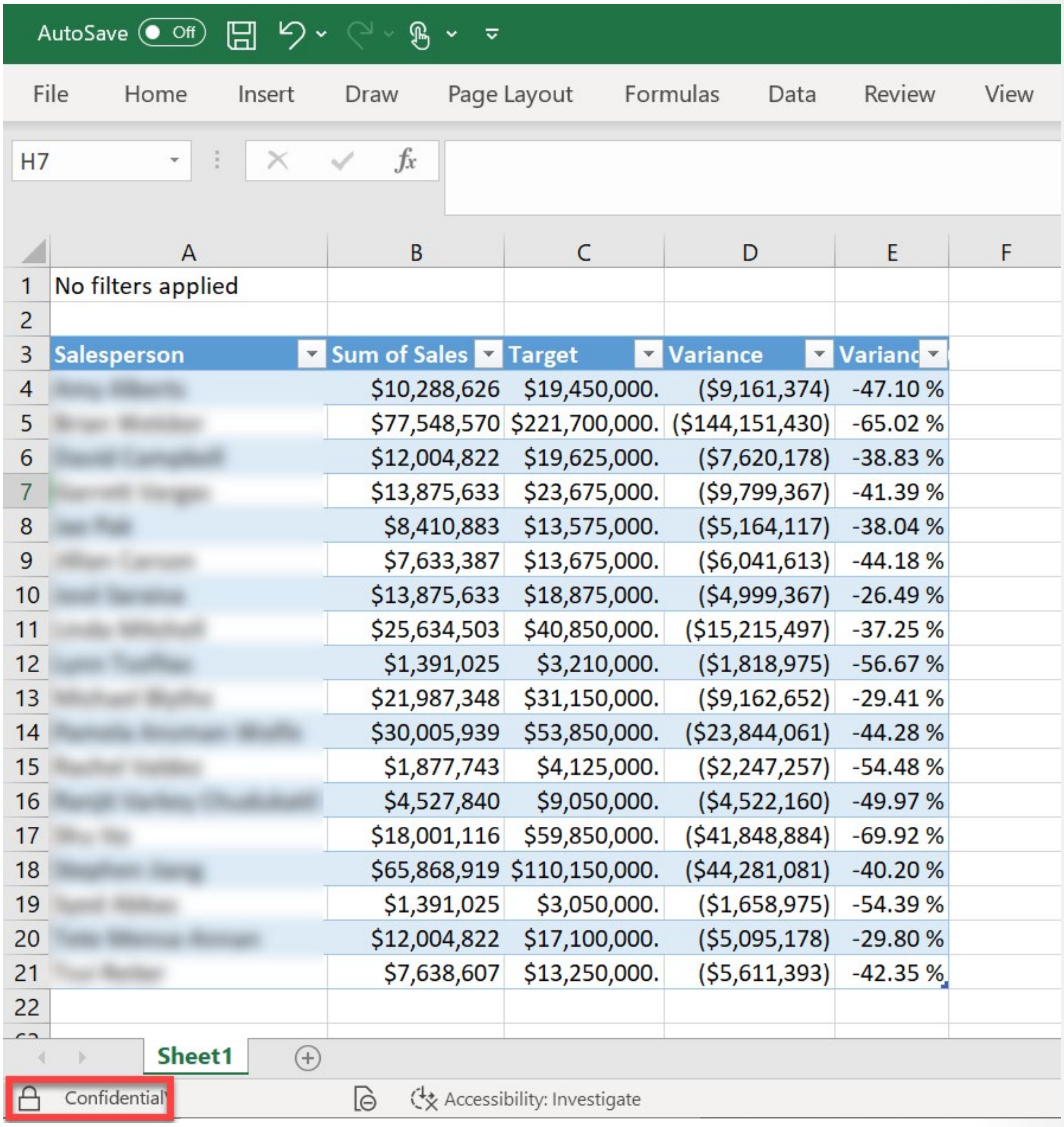


This will take you to a window, where you can assign a sensitivity label to your data. In our case, we have externally configured the following labels, and can now apply them to the data: **None, Personal, General**, **Confidential**, and **Highly confidential**. You can also go to **Microsoft 365 Security Center**[6] and define your own labels there.

Let's say I want to assign a **Confidential** label to our **Sales Data** report. When I change this label on the **Settings** pane, it now appears as a label on the report, as seen below.

This is crucial when exporting data. Data exported to Excel, PowerPoint, and PDF files will have sensitivity labels enforced. Suppose you wanted to export some data from **Sales Data** into an Excel file. If you are an authorized user, you will see the following Excel view when you export into Excel:

6 https://security.microsoft.com/homepage



	A	B	C	D	E	F
1	No filters applied					
2						
3	Salesperson	Sum of Sales	Target	Variance	Varianc	
4	[illegible]	$10,288,626	$19,450,000.	($9,161,374)	-47.10 %	
5	[illegible]	$77,548,570	$221,700,000.	($144,151,430)	-65.02 %	
6	[illegible]	$12,004,822	$19,625,000.	($7,620,178)	-38.83 %	
7	[illegible]	$13,875,633	$23,675,000.	($9,799,367)	-41.39 %	
8	[illegible]	$8,410,883	$13,575,000.	($5,164,117)	-38.04 %	
9	[illegible]	$7,633,387	$13,675,000.	($6,041,613)	-44.18 %	
10	[illegible]	$13,875,633	$18,875,000.	($4,999,367)	-26.49 %	
11	[illegible]	$25,634,503	$40,850,000.	($15,215,497)	-37.25 %	
12	[illegible]	$1,391,025	$3,210,000.	($1,818,975)	-56.67 %	
13	[illegible]	$21,987,348	$31,150,000.	($9,162,652)	-29.41 %	
14	[illegible]	$30,005,939	$53,850,000.	($23,844,061)	-44.28 %	
15	[illegible]	$1,877,743	$4,125,000.	($2,247,257)	-54.48 %	
16	[illegible]	$4,527,840	$9,050,000.	($4,522,160)	-49.97 %	
17	[illegible]	$18,001,116	$59,850,000.	($41,848,884)	-69.92 %	
18	[illegible]	$65,868,919	$110,150,000.	($44,281,081)	-40.20 %	
19	[illegible]	$1,391,025	$3,050,000.	($1,658,975)	-54.39 %	
20	[illegible]	$12,004,822	$17,100,000.	($5,095,178)	-29.80 %	
21	[illegible]	$7,638,607	$13,250,000.	($5,611,393)	-42.35 %	
22						

However, if you did not have permission you would be denied access to see the data, which ensures that only the appropriate users have access to view the data, making sure your data is secured.

If you are interested in learning more, please refer to **Apply Data Sensitivity Labels in Power BI**[7].

Module Review

Workspaces are a crucial feature of Power BI, allowing you to share reports, build dashboards, and collaborate amongst your teams. Through the past several discussions, we have described various tasks in a Power BI workspace that can increase performance in your reports, ensure appropriate security requirements are applied, make it easier to share content, and more. With this background you can add to your toolkit the ability to manage workspaces in Power BI Service so that you can build out your dashboards in

7 https://docs.microsoft.com/power-bi/collaborate-share/service-security-apply-data-sensitivity-labels

the efficient way possible. If you are interested in learning more, please refer to **Organize Work in the new Workspaces.**[8].

Knowledge Check

Question 1

What feature in Power BI service can you use to troubleshoot the flow of data from its source to destination?

- ☐ Usage metrics report
- ☐ Query caching
- ☐ Quick insights
- ☐ Lineage view

Question 2

A key tenant of data protection is sensitivity labels, which specifies what?

- ☐ The classification of critical content in Power BI
- ☐ Access to content in the Power BI service

8 https://docs.microsoft.com/power-bi/collaborate-share/service-new-workspaces

Answers

Question 1

How does the admin workspace role differ from other types of workspace roles?

- ☐ Admin is the only role that can publish or update apps.
- ■ Admins are the only role that can remove users.
- ☐ Admin is the only role that can create, edit, or delete content in a workspace.
- ☐ Admin is the only role that can publish content to a workspace.

Question 2

What is the best description of a workspace?

- ☐ A workspace is a feature in Power BI service that allows you to view reports only.
- ☐ A workspace is a feature that allows you to view and edit the data model, build visualizations, and transform the data.
- ☐ A workspace is a feature of Power BI Desktop that allows you to build reports only.
- ■ A workspace is a centralized location or repository that allows you to collaborate with colleagues and teams to create collections of reports, dashboards, etc.

Question 1

What feature in Power BI service can you use to troubleshoot the flow of data from its source to destination?

- ☐ Usage metrics report
- ☐ Query caching
- ☐ Quick insights
- ■ Lineage view

Question 2

A key tenant of data protection is sensitivity labels, which specifies what?

- ■ The classification of critical content in Power BI
- ☐ Access to content in the Power BI service

Module 11 Manage Datasets in Power BI

Parameters

Introduction to datasets and parameters

When your datasets are published to your organization's workspace in Power BI Service, everyone who needs access to those datasets can find them in a central location, and this provides opportunities for collaboration between teams. It also reduces the duplication of effort, as one dataset can be used by multiple users for different business reasons. For instance, one dataset can be used to create multiple Power BI reports. Since preparing and cleaning data can be so time consuming, sharing datasets can be a huge productivity boost for report authors.

This sharing of datasets needs to be actively managed for optimal organizational performance. For example, you can automate the refresh process, so that it becomes more efficient and users always have access to the latest data. You can also promote certain datasets over others, so users can clearly identify the best datasets to use.

The management of datasets also involves the implementation of clever parameters within those datasets, to aid decision making and solve business problems. For example, you can use parameters change the server or database name of your dataset, or a file path for a data source. You can also use parameters to configure incremental refreshes of your data, and to run 'what-if' scenarios and conduct scenario type analysis on the data.

Another key area of dataset management is setting up and maintaining a gateway, so you and other users can access your on-premises data source from the cloud. You also need to prepare for potential issues that might arise regarding this gateway, which could interrupt user access to the datasets. The effect of a service connectivity issue could be detrimental to productivity of your users, if they cannot access the data, they cannot do their jobs, and the organization's decision-making capability is at a standstill. Being prepared to deal with such issues in a timely manner is critical.

Imagine you work as a Power BI Developer for Tailwind Traders and have created reports for multiple teams across the organization. But your work is not done, you've been asked by the report end users if you can make the reports more dynamic, so they can filter the data themselves, and if you find a way for them to run what-if scenarios. Management has also requested that you take action to guarantee the coherency and integrity of your datasets. They want the datasets available in one place, for future use, and they want you to automate the refresh process, to ensure the data is kept up to date.

By the end of this module you'll be able to:

- Create dynamic reports with parameters
- Create what-if parameters
- Use a Power BI gateway to connect to on-premise data sources
- Configure a dataset scheduled refresh
- Configure incremental refresh settings
- Manage and promote datasets
- Troubleshoot service connectivity
- Boost performance with query caching (Premium)

Dynamic reports with parameters

Dynamic reports are reports in which the data can be changed by a developer, according to user specifications. Dynamic reports are valuable, as a single report can be used for multiple purposes. If you use dynamic reports, you'll have fewer individual reports to create, which will save organizational time and resources.

You can use parameters by determining the values that you want to see data for in the report, and the report updates accordingly by filtering the data for you.

Creating dynamic reports allows you to give end-users more power over the data that is displayed in your reports; they can change the data source and filtering the data by themselves.

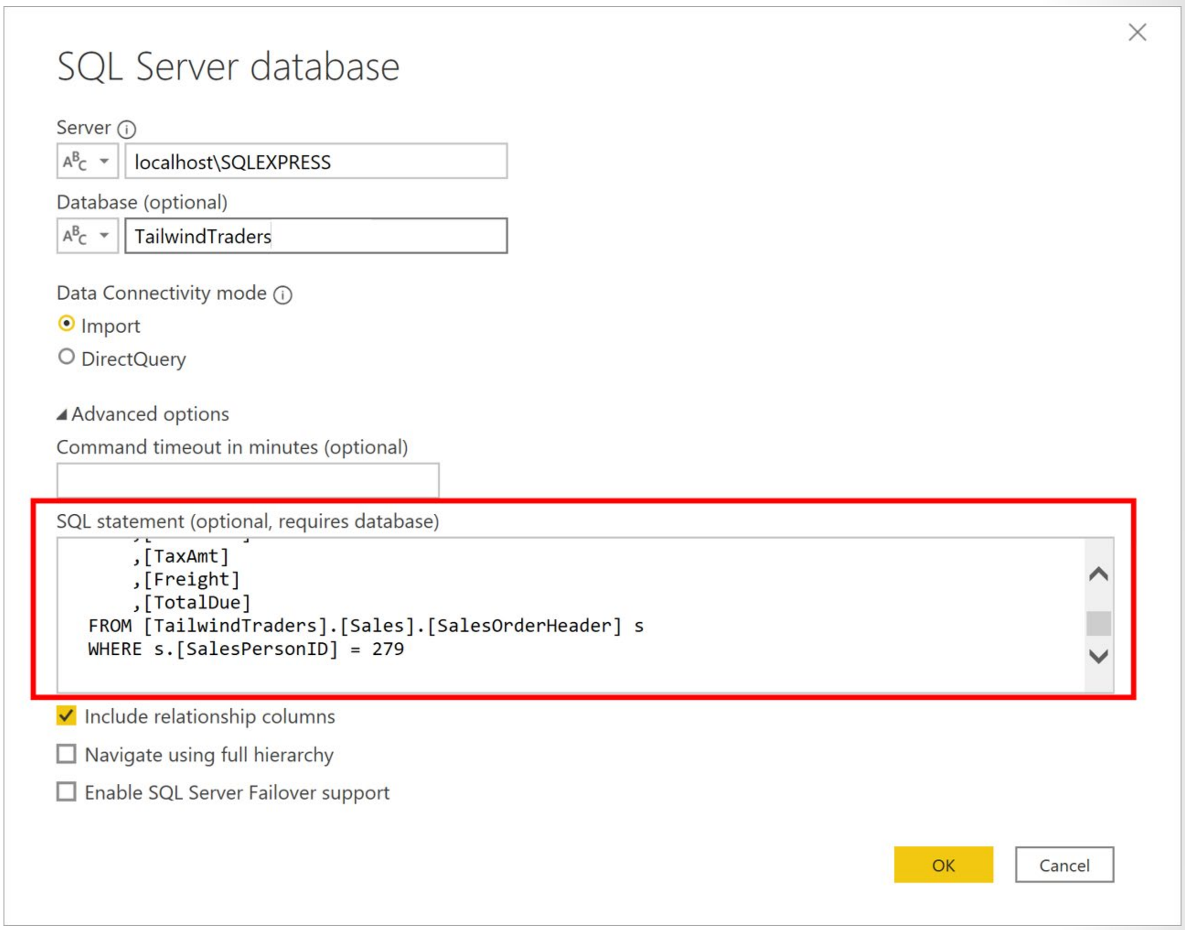
In this example, you've created a report for the Sales team that displays the sales data in the SQL Server database. The report gives a holistic view of how the Sales team is performing. Whilst the report is extremely useful, the Sales team members would like to be able to filter the report, so they can view only their own data and more easily track their performance against their sales targets.

Create dynamic reports for individual values

To create a dynamic report, you first need to write your SQL query and then you need to use the **Get data** feature in Power BI Desktop to connect to the database.

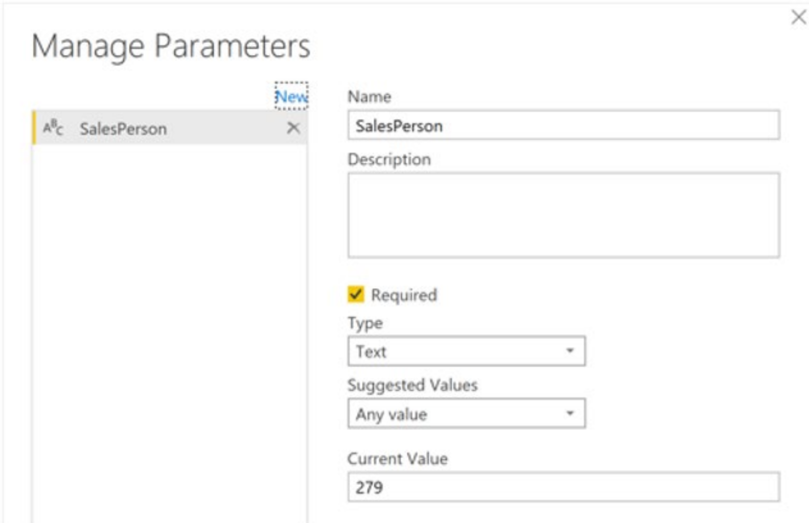
In this example, you connect to your database on SQL Server. In the **SQL Server database** window, after you enter your server details, select the **Advanced options**, then paste the SQL query into the **SQL statement** box, and then select **OK**.

```sql
SELECT [SalesOrderID]
      ,[OrderDate]
      ,[OnlineOrderFlag]
      ,[SalesOrderNumber]
      ,[CustomerID]
      ,[SalesPersonID]
      ,[TerritoryID]
      ,[SubTotal]
      ,[TaxAmt]
      ,[Freight]
      ,[TotalDue]
  FROM [TailwindTraders].[Sales].[SalesOrderHeader] s
  WHERE s.[SalesPersonID] = 279
```

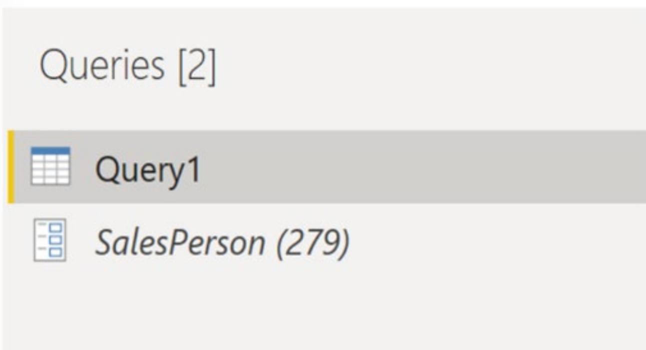


When the connection is made, you will see the data in the preview window. Select **Edit** to open the data in Power Query Editor.

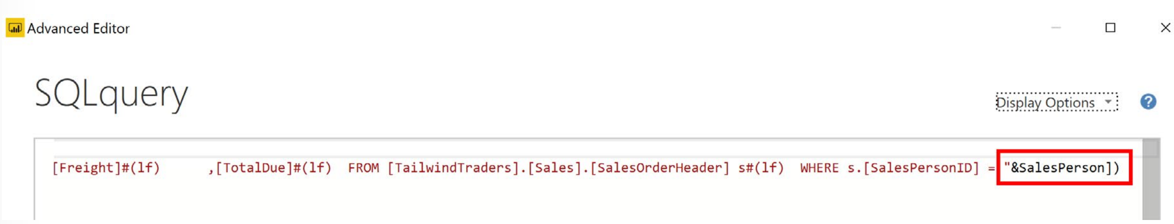
The next step is to create the parameter. On **Home** tab, select **Manage parameters** > **New parameter**. On the **Parameters** window, change the default parameter name to something more descriptive, so its purpose is clear. In this case, you change the name to *SalesPerson*. Select **Text** from the **Type** list and **Any value** from the **Suggested value** list, then select **OK**.



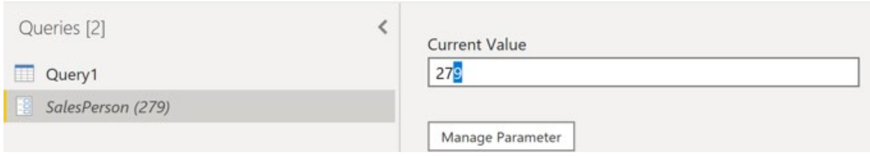
You'll then see a new query for the parameter you created.



Now you need to adjust the code in SQL query, to take account of your new parameter. Right-click **Query1** and select **Advanced editor**, then replace the existing value in the execute statement with an ampersand (&) followed by your parameter name (**SalesPerson**), as illustrated in the following image. Ensure there are no errors at bottom of the window, then select **Done**.

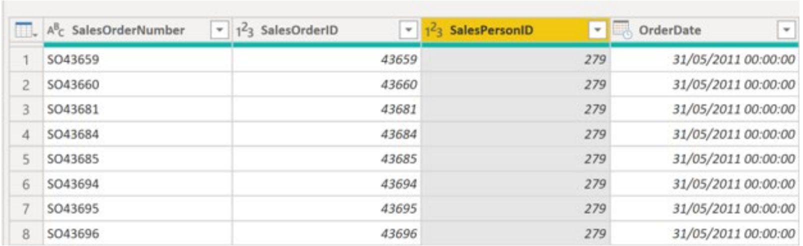


You won't see anything different, but Power BI will have executed the query. To confirm that is the case, you can run a test. Select the parameter query, then enter a new value into the **Current Value** box.



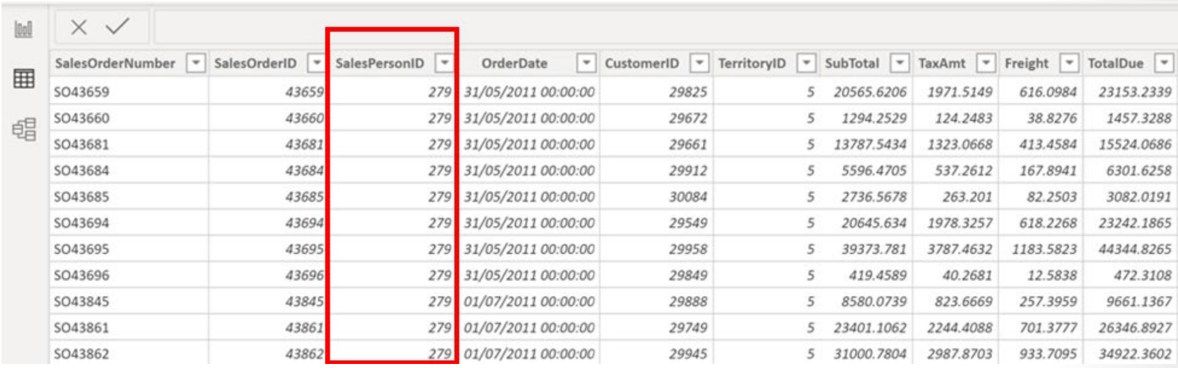
You might see a warning icon displaying next to the query. If that is the case, select that query to view the warning message, which says that permission is required to run this native database query. Select **Edit Permission**, then select **Run**.

When the query executes successfully, you'll see the parameter updates and displays the new value.



	SalesOrderNumber	SalesOrderID	SalesPersonID	OrderDate
1	SO43659	43659	279	31/05/2011 00:00:00
2	SO43660	43660	279	31/05/2011 00:00:00
3	SO43681	43681	279	31/05/2011 00:00:00
4	SO43684	43684	279	31/05/2011 00:00:00
5	SO43685	43685	279	31/05/2011 00:00:00
6	SO43694	43694	279	31/05/2011 00:00:00
7	SO43695	43695	279	31/05/2011 00:00:00
8	SO43696	43696	279	31/05/2011 00:00:00

Select **Close and Apply** to return to the report editor. Now you can apply the parameter to the report. Select **Edit queries** > **Edit parameters**, then on **Edit Parameters** window, enter a new value and select **OK**. Then select **Apply changes** and run the native query again. Now when you view the data, you'll see the data for the new value that was passed through the parameter.



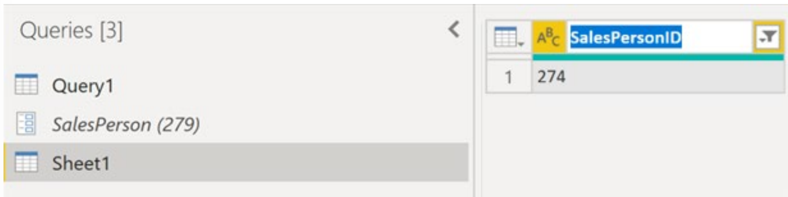
SalesOrderNumber	SalesOrderID	SalesPersonID	OrderDate	CustomerID	TerritoryID	SubTotal	TaxAmt	Freight	TotalDue
SO43659	43659	279	31/05/2011 00:00:00	29825	5	20565.6206	1971.5149	616.0984	23153.2339
SO43660	43660	279	31/05/2011 00:00:00	29672	5	1294.2529	124.2483	38.8276	1457.3288
SO43681	43681	279	31/05/2011 00:00:00	29661	5	13787.5434	1323.0668	413.4584	15524.0686
SO43684	43684	279	31/05/2011 00:00:00	29912	5	5596.4705	537.2612	167.8941	6301.6258
SO43685	43685	279	31/05/2011 00:00:00	30084	5	2736.5678	263.201	82.2503	3082.0191
SO43694	43694	279	31/05/2011 00:00:00	29549	5	20645.634	1978.3257	618.2268	23242.1865
SO43695	43695	279	31/05/2011 00:00:00	29958	5	39373.781	3787.4632	1183.5823	44344.8265
SO43696	43696	279	31/05/2011 00:00:00	29849	5	419.4589	40.2681	12.5838	472.3108
SO43845	43845	279	01/07/2011 00:00:00	29888	5	8580.0739	823.6669	257.3959	9661.1367
SO43861	43861	279	01/07/2011 00:00:00	29749	5	23401.1062	2244.4088	701.3777	26346.8927
SO43862	43862	279	01/07/2011 00:00:00	29945	5	31000.7804	2987.8703	933.7095	34922.3602

You can now create a report that displays data for one particular value at a time. If you want to display data for multiple values at the same time, you need to carry out some additional steps, as outlined in the next section.

Create dynamic reports for multiple values

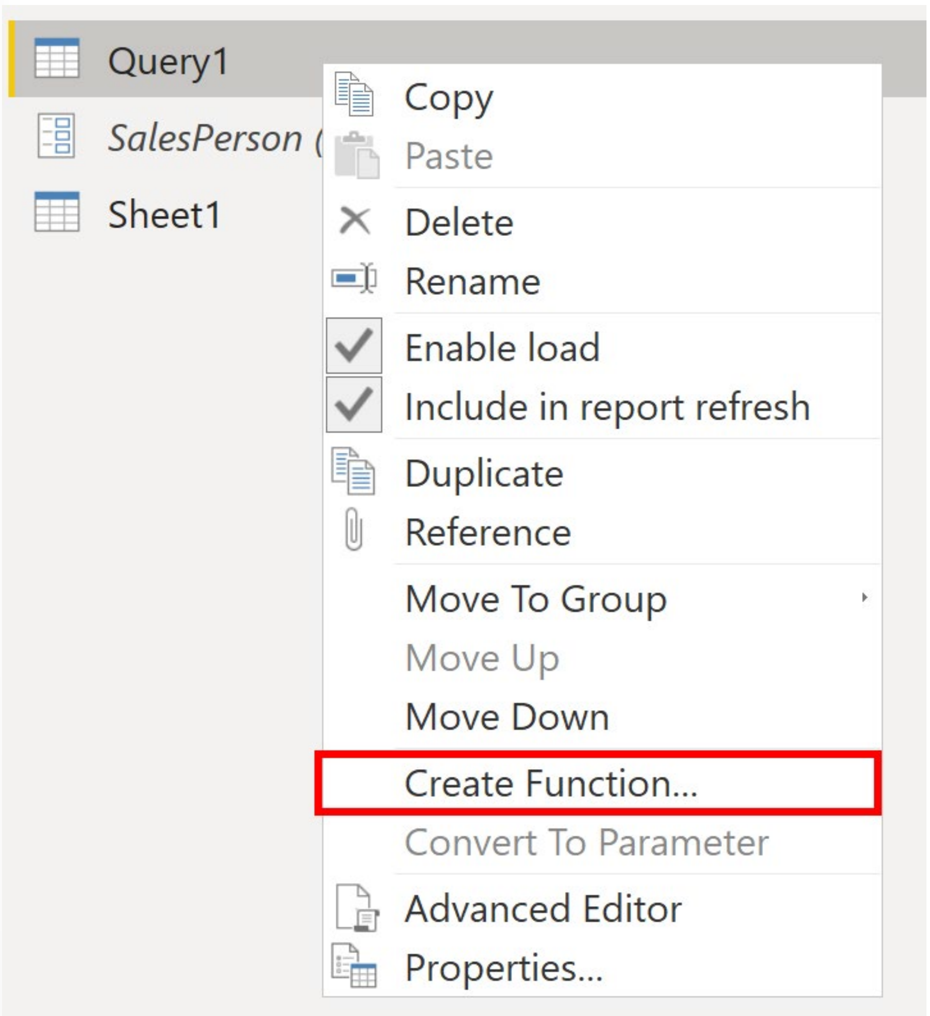
To cater for multiple values at a time, you first need to create an Excel worksheet that has a table consisting of one column, which contains the list of values.

Next, use the **Get data** feature in Power BI Desktop to connect to the data in that Excel worksheet, and on the **Navigator** window, select **Edit** to open the data in Power Query Editor, where you'll see a new query for the data table.



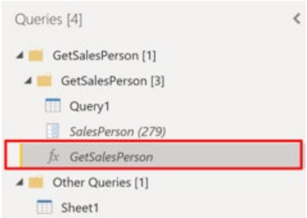
Rename the column in the table to something more descriptive, then change the column data type to text, so that it matches the parameter type and you avoid any data conversion problems. In the query **Properties** section, also change the name of the data source to something more descriptive – *SalesPersonID* in this case.

Next, you need to create a function that'll pass the new **SalesPersonID** query into the **Query1**. Right-click **Query1** and select **Create function**.

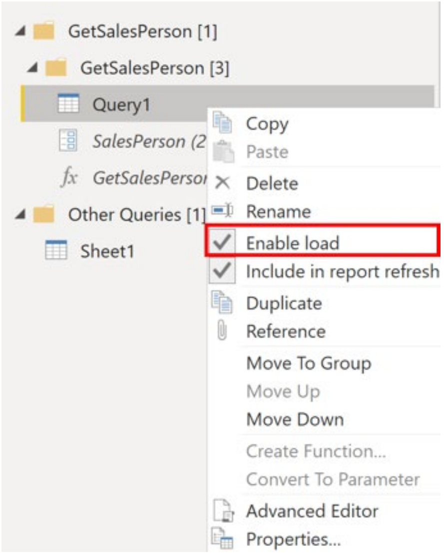


Enter a name for the function and select **OK**.

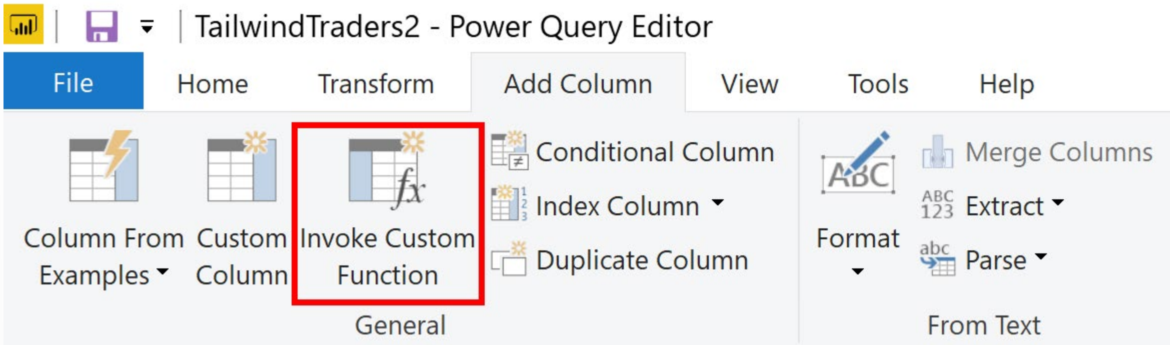
You'll then see your new function in the **Queries** pane.



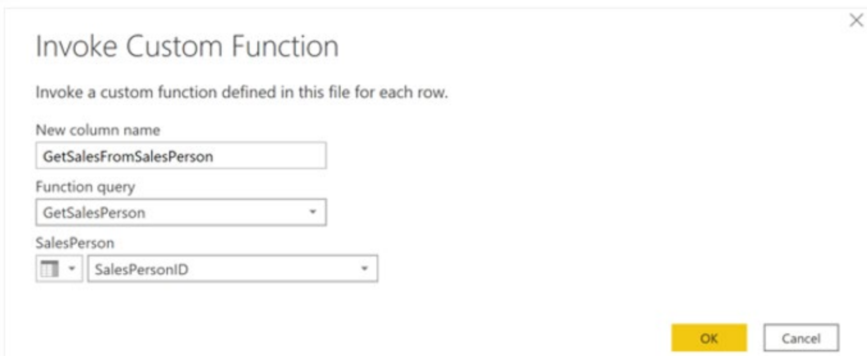
To ensure **Query1** doesn't show up in the field list for the report, which could potentially confuse end-users, you can disable it loading in the report. Right-click **Query1** again, then select **Enable load** (selected by default) to disable the feature.



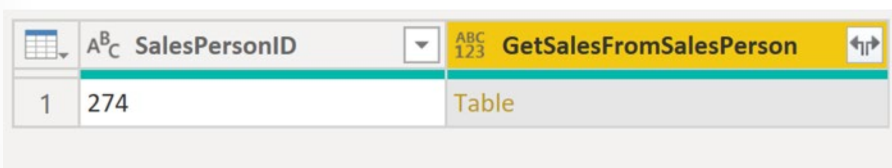
Select the **SalesPerson** query you loaded from the Excel worksheet, then on the **Add Column** tab, select **Invoke custom function** to run the custom function you just created.



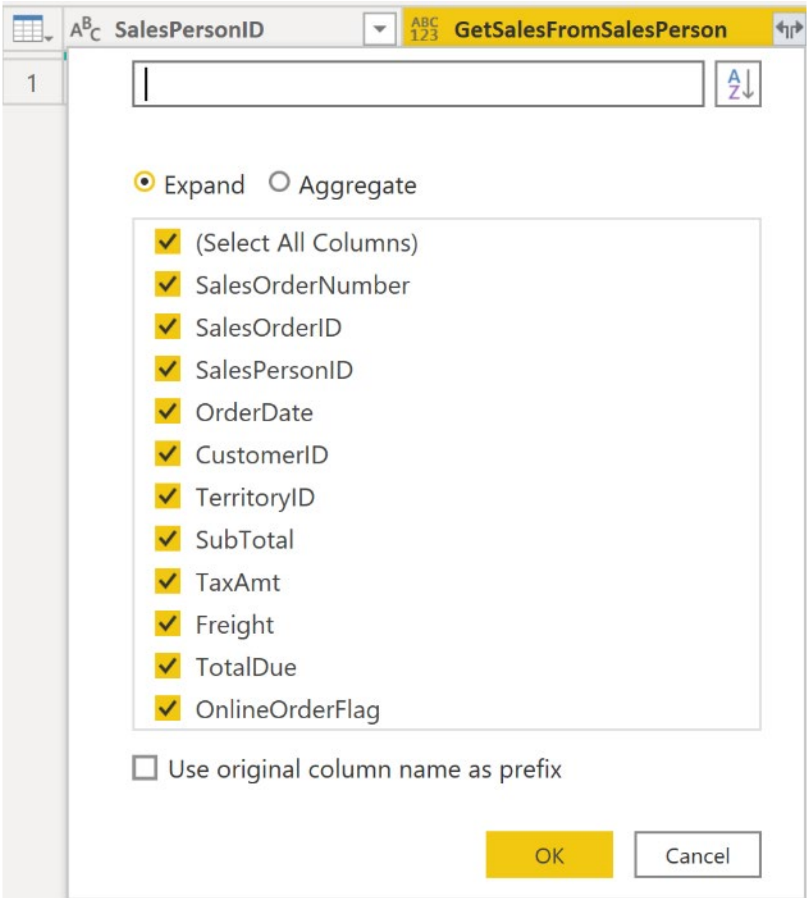
On the **Invoke Custom Function** window, select your function from **Function query** list. You'll see that the **New column name** updates automatically and the table that contains the values you're going to pass through the parameter is selected by default. Select **OK**, and if required, run the native query.



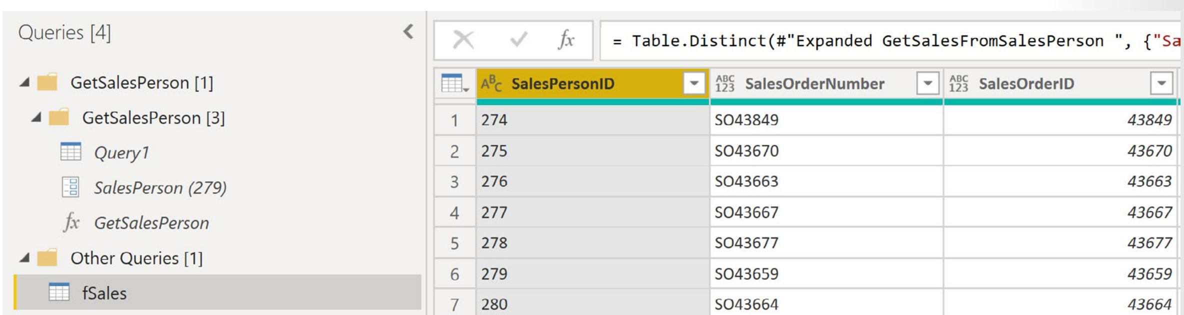
You'll then see a new column for your **GetSalesFromSalesPerson** function next to the **SalesPersonID** column.



Select the two arrows icon in that new column header, then select the check boxes of the columns that you want to load. This is where you determine the details that will be available in the report for each value (sales person ID). Clear the **Use original column name as prefix** checkbox at the bottom, as you do not need to see a prefix with the column names in the report. Then select **OK**.



Now you can see the data for the columns you selected, for each value (sales person ID).



If required, you can add more values (sales people IDs) to the **SalesPersonID** column in the Excel worksheet, or change the existing values. Save your changes, then go back to Power Query Editor. On the **Home** tab, select **Refresh Preview**, then run the native query again (if required), and you'll see the sales from the new sales people IDs that you added into the worksheet.

Click **Close and Apply** to return to the report editor, where you'll see the new column names in **Fields** pane and you can start building your report.

Knowledge Check

Question 2

What benefit does dynamic reports provide to end users?

☐ It provides static views of data that can only be manipulated by report developers.

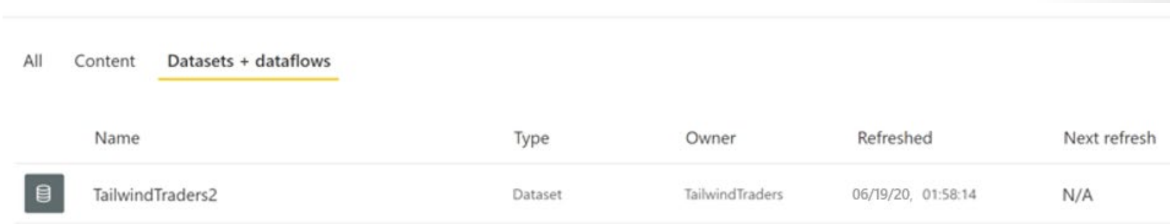
☐ It gives end users more control over the data that is displayed.

Datasets

Dataset scheduled refresh

The **Scheduled refresh** feature in Power BI Service allows you to define the frequency and time slots to refresh a particular dataset. Scheduling the refresh of your data will save you time, as you don't have to manually refresh the data. It also ensures that the end users can access the most up-to-date data.

In this example, you are creating a report but you realize that the version of the Sales data you are using isn't the most up to date. You check the refresh status and see that it was last refreshed 10 days ago! And there is no refresh scheduled to take place.



Considering how important it is to have accurate sales data, you need to find a solution. The data usually gets updated every week but you don't want to have to come back to the report every week to manually refresh the dataset, and sometimes you forget to do it. You decide to use Power BI's scheduled refresh functionality to solve this problem.

Set up a refresh schedule

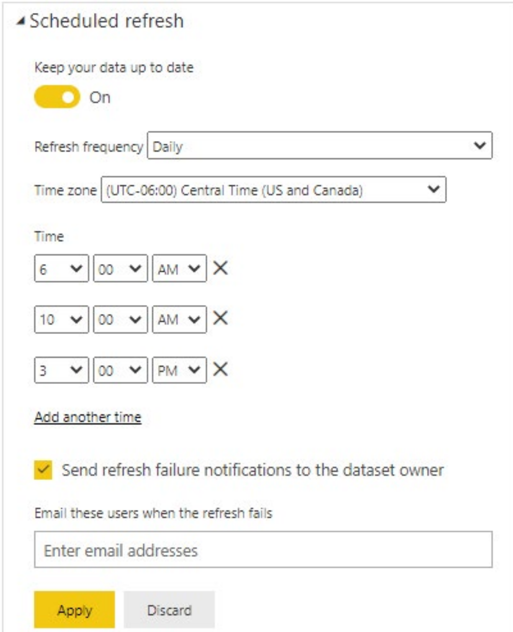
Before you can set up a refresh schedule, you must have created a gateway connection.

To set up a refresh schedule for your dataset, go to the **Datasets + dataflows** page. Hover over the dataset for which you want to set up the schedule then, the select the **Schedule refresh** icon.

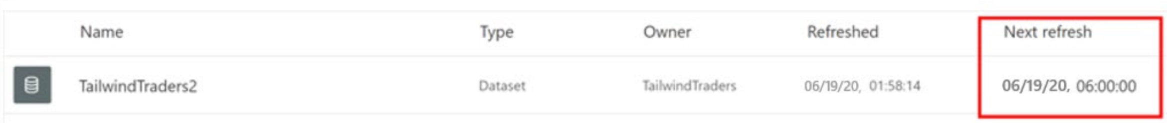
On the **Settings** page, turn on the scheduled refresh feature. Next, select the refresh frequency, and ensure the correct time zone is selected. You then add the time(s) that you want the refresh to occur. You can configure up to eight daily time slots, if your dataset is on shared capacity, or 48 time slots on Power BI Premium. When you have finished configuring the scheduled refresh, select **Apply**.

NOTE: Whilst you can set a time for the refresh, be aware that the refresh might not take place at that exact time. Power BI starts scheduled refreshes on a best effort basis. The target is to initiate the refresh within 15 minutes of the scheduled time slot, but a delay of up to one hour can occur if the service can't allocate the required resources sooner.

In this example, you want the system to refresh the Sales data on a daily basis 6:00 AM, 10:00 AM, and 3:00 PM, as illustrated in the following image.



When you have configured a refresh schedule, the dataset settings page informs you of the next refresh time, as you can see in the following image.

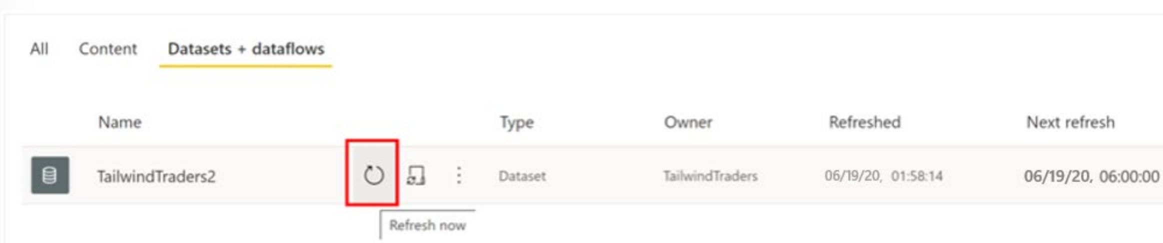


Perform an on-demand refresh

In addition to the scheduled refreshes, you can refresh a dataset at any time by performing an on-demand refresh. This type of refresh doesn't affect the next scheduled refresh time.

For example, you might need to refresh now because you need to view the most recent data and cannot wait for the next refresh time, or you might want to test your gateway and data source configuration.

To perform an on-demand refresh, on the **Datasets + dataflows** page, hover over the dataset that you want to refresh, then select the **Refresh now** icon.

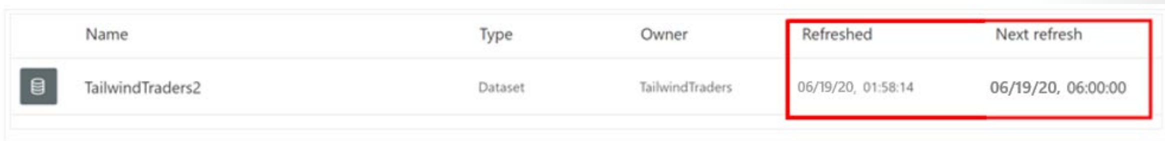


Check the refresh status and history

You can check the refresh status and history at any time. This is useful if you want to find out when the last refresh occurred and when the next one is scheduled. It is also good practice to check the status of your datasets from time to time, to see if there have been any refresh errors.

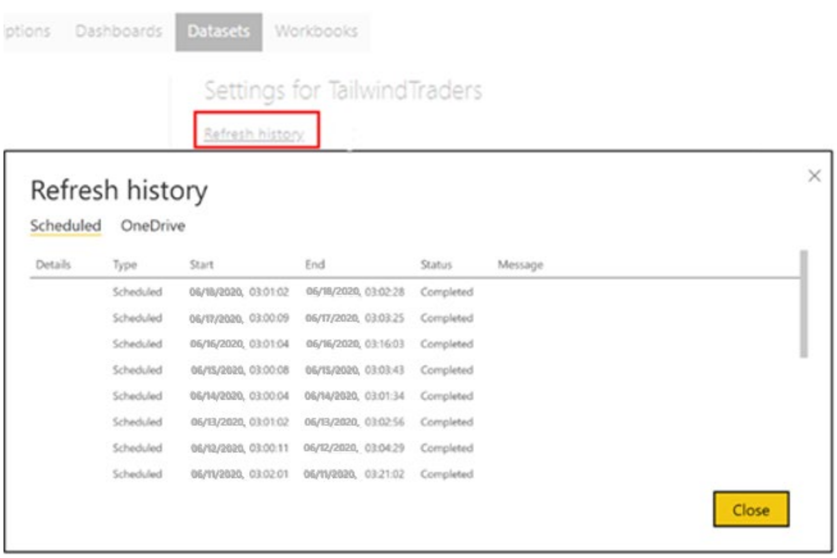
NOTE: Power BI deactivates your refresh schedule after four consecutive failures or when the service detects an unrecoverable error that requires a configuration update, such as invalid or expired credentials. It is not possible to change the consecutive failures threshold.

A quick way to check the refresh status is to view the list of datasets in a workspace.



If a dataset displays a small warning icon, you'll know that the dataset is currently experiencing an issue. Select the warning icon to get more information.

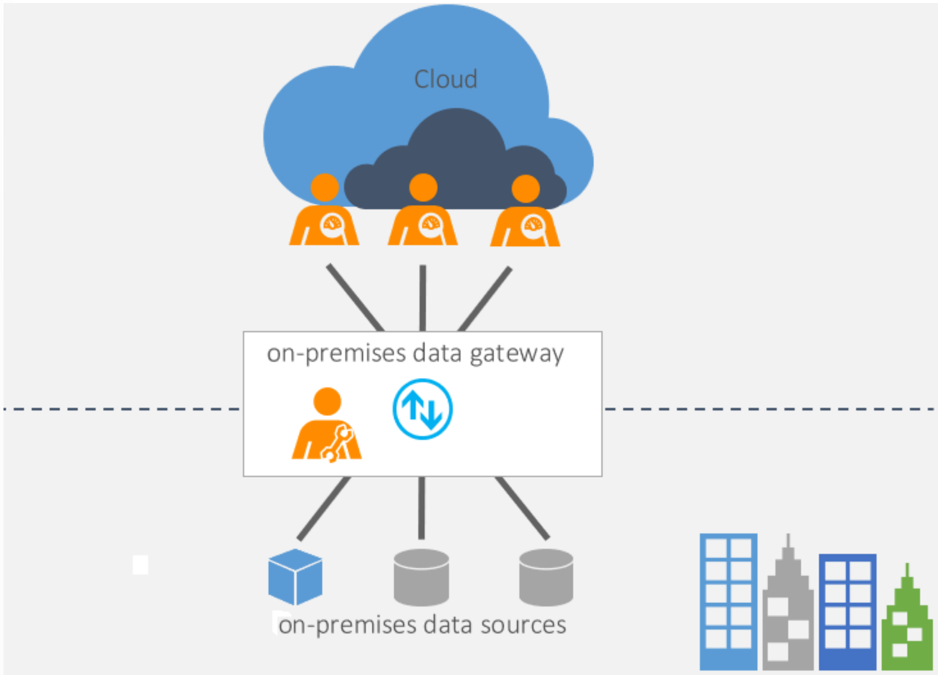
You should also check the refresh history occasionally, to review the success or failure status of past synchronization cycles. To view the refresh history, open the dataset's settings page, then select **Refresh history**.



Troubleshoot service connectivity

Gateway software acts like a bridge - it allows organizations to retain databases and other data sources on their on-premises networks, and access that on-premises data in cloud services, such as Power BI and Azure Analysis Services.

A gateway facilitates quick and secure behind-the-scenes communication flowing from a user in the cloud to your on-premises data source, and then back again to the cloud.



There are two types of on-premises gateway:

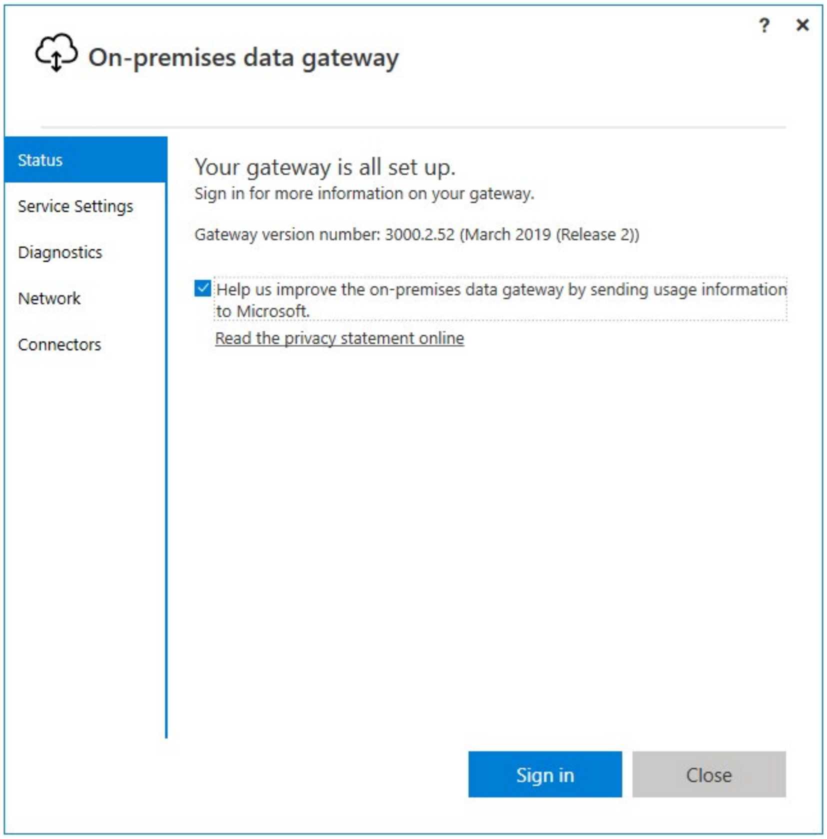
- Organization mode - Allows multiple users to connect to multiple on-premises data sources and is suitable for complex scenarios.
- Personal mode - Allows one user to connect to data sources. This type of gateway can be used only with Power BI and it can't be shared with other users, so it is suitable in situations where you're the only one in your organization who creates reports. You install the gateway on your local computer, which needs to stay online in order for the gateway to work.

Use on-premises gateway

Before you can connect to your on-premises data source, you need to **install the on-premises data gateway**[1], then configure it to suit your organizational needs. This task is usually done by an admin in your organization.

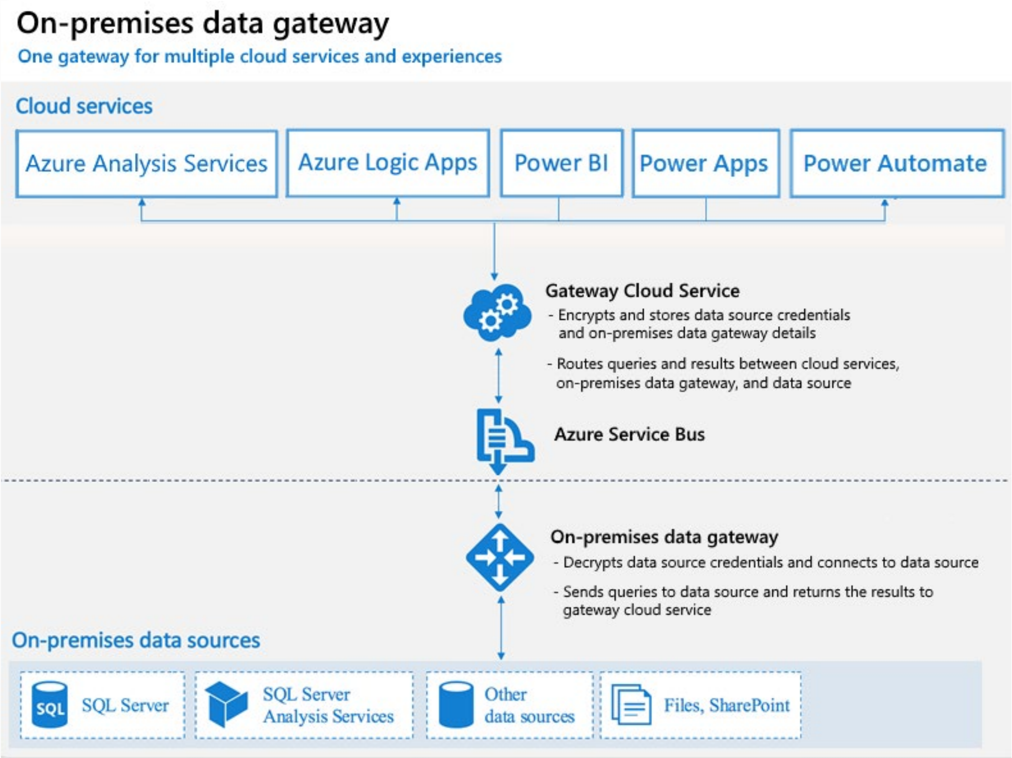
When the on-premises gateway is installed and configured, you can start the gateway and sign in using your Office 365 organization account.

1 https://docs.microsoft.com/data-integration/gateway/service-gateway-install



When you are working in the cloud and interact with an element that is connected to an on-premises data source, the following actions occur:

- The cloud service creates a query and the encrypted credentials for the on-premises data source. The query and credentials are sent to the gateway queue for processing.
- The gateway cloud service analyzes the query and pushes the request to Azure Service Bus.
- Azure Service Bus sends the pending requests to the gateway.
- The gateway gets the query, decrypts the credentials, and connects to one or more data sources with those credentials.
- The gateway sends the query to the data source to be run.
- The results are sent from the data source back to the gateway and then to the cloud service. The service then uses the results.



Troubleshoot on-premises data gateway

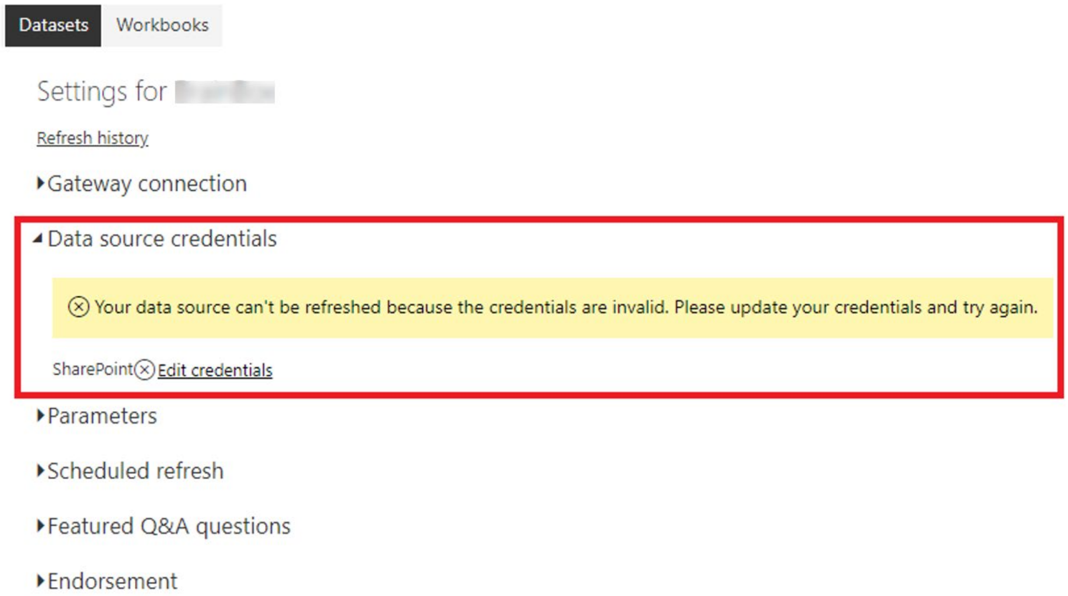
Troubleshooting a gateway is an ever-changing topic. Please looks at these documents for the latest troubleshooting guidance:

- To learn how to run a network port test, see **Adjust communication settings for the on-premises data gateway**[2].
- To get information on how to provide proxy information for your gateway, see **Configure proxy settings for the on-premises data gateway**[3].
- To find the current data center region you're in, see **Set the data center region**[4].

Cloud services like SharePoint Online do not require any gateway, as the data is already in the cloud. You only need to provide your authorization credentials to setup a data source connection.

If your report fails to refresh, ensure that your data source credentials are up to date.

2 https://docs.microsoft.com/data-integration/gateway/service-gateway-communication#network-ports-test
3 https://docs.microsoft.com/data-integration/gateway/service-gateway-proxy
4 https://docs.microsoft.com/data-integration/gateway/service-gateway-data-region



If your data source credentials are up to date. You'll need to take further action to investigate and resolve the issue.

For information regarding the different scenarios you might face when refreshing data within the Power BI service, see **Troubleshooting refresh scenarios**[5].

Module Review

You took advantage of Power BI's great features to manage your datasets. You started this task by using parameters that are stored in a Microsoft Excel workbook to create dynamic reports in Power BI Service, and thereby give end users the ability to filter the data for specific values. Next, you used data refresh to automate the refresh process and make it more efficient. Lastly, you became familiar with how the on-premises gateway and got some ideas on how to troubleshoot potential connectivity issues.

These dataset management techniques will help you to increase the ease of access and up-to-date nature of your datasets - and allow you to build high-quality reports and dashboards, so your users can make real-time decisions.

Knowledge Check

Question 1

Where are the dataset-scheduled refreshes configured?

- ☐ Power BI Desktop
- ☐ Power BI Service
- ☐ AppSource

5 https://docs.microsoft.com/power-bi/connect-data/refresh-troubleshooting-refresh-scenarios

Answers

Question 2

What benefit does dynamic reports provide to end users?

□ It provides static views of data that can only be manipulated by report developers.

■ It gives end users more control over the data that is displayed.

Question 1

Where are the dataset-scheduled refreshes configured?

□ Power BI Desktop

■ Power BI Service

□ AppSource

Module 12 Row-level Security

Security in Power BI

Security overview in Power BI

In Power BI, you can secure reports and workspaces by sharing them to active directory users and groups. It is also possible to share a single report, but have users see different data, according to their job role.

As an example, imagine you work for Tailspin Traders. You have the following table to track your sales:

customername	empID	department	product	price	quantity	orderAmount
Kelli Hinojos	1	Game	Settlers of Air	1	24.99	24.99
Jeffrey Reiss	5	Sports	Driver - Stiff Shaft	1	399.99	399.99
Roselyn James	7	Clothing	V-Neck T-Shirt	1	19.99	19.99
Lavonna Domingo	5	Sports	Golf Balls - Dual Core	3	32.5	97.5
Hermina Leslie	7	Clothing	Athletic Shorts	4	17.75	71
Jess Dammann	6	Automotive	Tire Guard	1	44.99	44.99
Kitty Hudman	1	Game	Santo Domingo	1	31	31
Sonia Coss	9	Clothing	Leather Sandels	2	111.97	223.94
Becky Pearsall	6	Automotive	True Coat	1	980	980
Echo Lundeen	3	Sports	Frisbee Golf Set	2	98	196
Sheryl Cayton	9	Clothing	Hoodie	1	44.99	44.99
Veronika Lopes	2	Automotive	Window Scrape	4	9.99	39.96
Sally Corliss	8	Game	Lords of Avalon	1	19.99	19.99
Sheldon Allende	3	Sports	Putting Green	1	59.25	59.25
Noma Yoakum	7	Clothing	Bathing Trunks	1	42.99	42.99
Hazel Kapinos	5	Sports	Weighted Bands	2	8.99	17.98
Francisco Ayers	7	Clothing	Hoodie	1	44.99	44.99
Tom Etienne	4	Game	Spider, spider	1	44.5	44.5
Jamel Carol	5	Sports	Boxing gloves	1	99	99
Rosemary Treacy	4	Game	Invest in it All	1	31.99	31.99

You also have the following table for employee information:

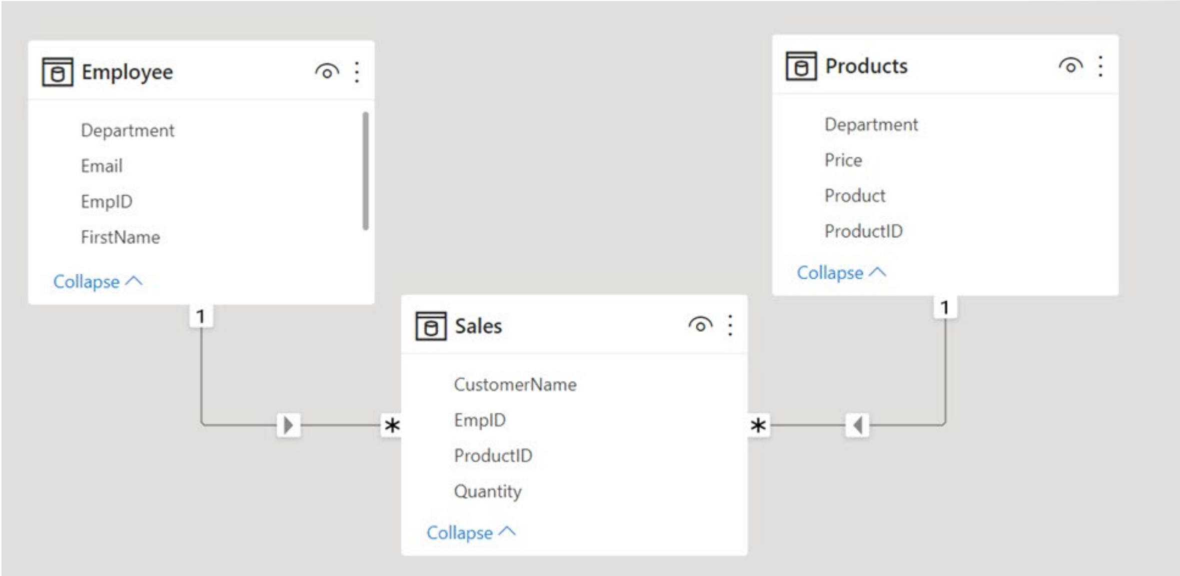
customername	empID	productID	quantity
Kelli Hinojos	1	82	1
Jeffrey Reiss	5	24	1
Roselyn James	7	67	1
Lavonna Domingo	5	42	3
Hermina Leslie	7	88	4
Jess Dammann	6	19	1
Kitty Hudman	1	47	1
Sonia Coss	9	98	2
Becky Pearsall	6	73	1
Echo Lundeen	3	3	2
Sheryl Cayton	9	58	1
Veronika Lopes	2	8	4
Sally Corliss	8	61	1
Sheldon Allende	3	91	1
Noma Yoakum	7	16	1
Hazel Kapinos	5	4	2
Francisco Ayers	7	71	1
Tom Etienne	4	36	1
Jamel Carol	5	83	1
Rosemary Treacy	4	65	1

This is the products table:

productID	department	product	price
3	Sports	Frisbee Golf Set	98
4	Sports	Weighted Bands	8.99
8	Automotive	Window Scrape	9.99
16	Clothing	Bathing Trunks	42.99
19	Automotive	Tire Guard	44.99
24	Sports	Driver - Stiff Shaft	399.99
36	Game	Spider, spider	44.5
42	Sports	Golf Balls - Dual Core	32.5
47	Game	Santo Domingo	31
58	Clothing	Hoodie	44.99
61	Game	Lords of Avalon	19.99
65	Game	Invest in it All	31.99
67	Clothing	V-Neck T-Shirt	19.99
71	Clothing	Hoodie	44.99
73	Automotive	True Coat	980
82	Game	Settlers of Air	24.99
83	Sports	Boxing gloves	99
88	Clothing	Athletic Shorts	17.75
91	Sports	Putting Green	59.25
98	Clothing	Leather Sandels	111.97

You would like to make one report where employees in a specific department can only see the sales for that department. For instance, Maria Cameron works in the Game department and should only see the sales from that department. She should not see the sales from Sports, Clothing, or Automotive.

This data is organized in a star schema. The sales table has all of the attributes of a fact table, while employees and products are dimension tables. The data model is in the following screenshot:



There are two ways to implement row-level security in Power BI: the static method and the dynamic method.

Row-level security (RLS) uses a DAX filter as the core logic mechanism. This module will demonstrate how you can implement row-level security in Power BI using DAX to ensure that only the appropriate person is seeing the appropriate records.

Static Method

The static method in Row-level security (RLS) uses a fixed value in the DAX filter, while the dynamic method uses a DAX function. You will see this in action shortly.

Row-level security (RLS) takes several steps to configure. We will perform them in this order:

1. Create a report in Power BI Desktop.
   - Import the data.
   - Confirm the data model between both tables.
   - Create the report visuals.
2. Create RLS roles in Power BI Desktop using DAX.
3. Test the roles in Power BI Desktop.
4. Deploy the report to the Power BI service.
5. Add members to the role in the Power BI service.
6. Test the roles in the Power BI Service.

Create a report in Power BI Desktop

First, we follow the typical steps to create a report in Power BI Desktop. We use Power Query to retrieve and clean the data. Confirm that the relationship exists between the two tables using the modeling tab. It should be a 1 to many relationship on the empID column.

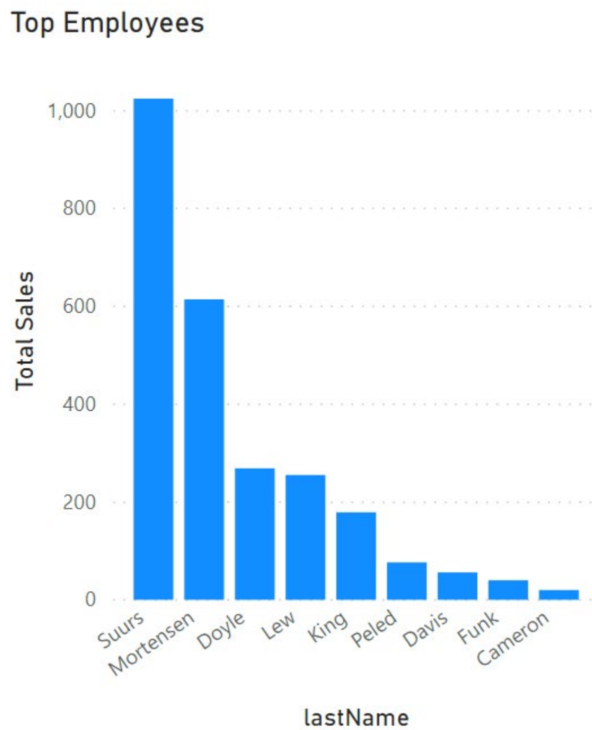
Then we create a Power BI report.

2.54K

Total Sales

Top Products Sales

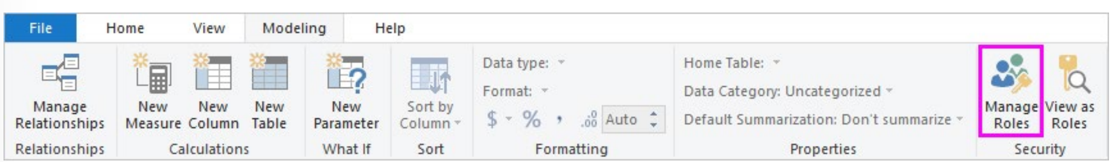
product	department	Total Sales
True Coat	Automotive	980.00
Driver - Stiff Shaft	Sports	399.99
Leather Sandels	Clothing	223.94
Frisbee Golf Set	Sports	196.00
Boxing gloves	Sports	99.00
Golf Balls - Dual Core	Sports	97.50
Hoodie	Clothing	89.98
Athletic Shorts	Clothing	71.00
Putting Green	Sports	59.25
Tire Guard	Automotive	44.99
Spider, spider	Game	44.50
Bathing Trunks	Clothing	42.99
Window Scrape	Automotive	39.96
Invest in it All	Game	31.99
Santo Domingo	Game	31.00
Settlers of Air	Game	24.99
Lords of Avalon	Game	19.99
V-Neck T-Shirt	Clothing	19.99
Weighted Bands	Sports	17.98
Total		**2,535.04**



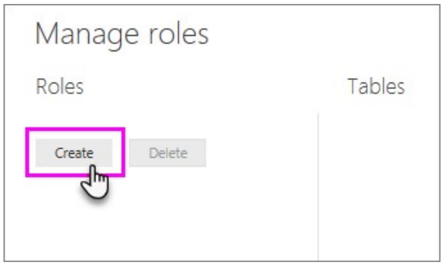
Notice how the table has rows for all of our sales, including all of the departments. We will limit this so that only employees of a specific department can see their own sales.

Create RLS roles in Power BI Desktop using DAX

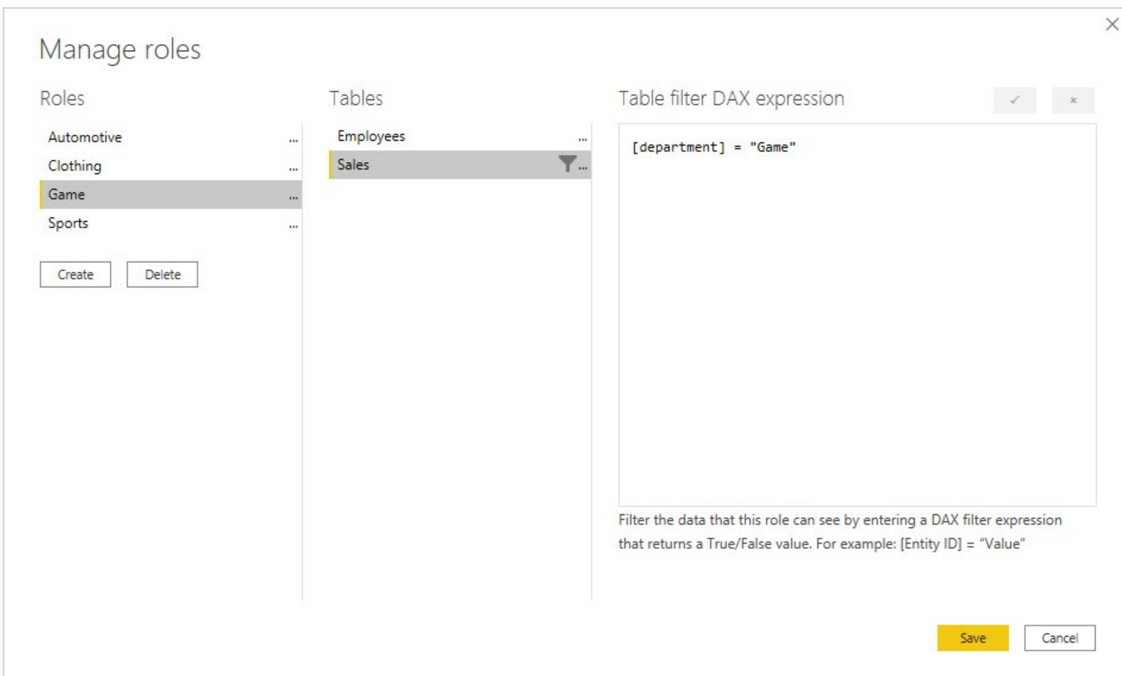
Select the **Modeling** tab, click **Manage Roles**.



From **Manage Roles**, click **Create**.



Power BI Row-level security (RLS) uses DAX to control who can see which data. Think of it as always adding another filter to the appropriate users, no matter what filters, slicers, or interactions they choose on a Power BI Report. Here we will create a role for each department and add a DAX expression to it. For instance, we will create a role called “Game”. We will add the DAX expression [department] = "Game". Each time a member of that role interacts with the report, Power BI will add that filter to their interactions, thus limiting them what they see. Notice how you use a fixed value in the filter on the right-side of the equal sign, in this case, “Game”. This means that if you ever need to add a category, you would need to create a new role with a new value in the DAX expression.



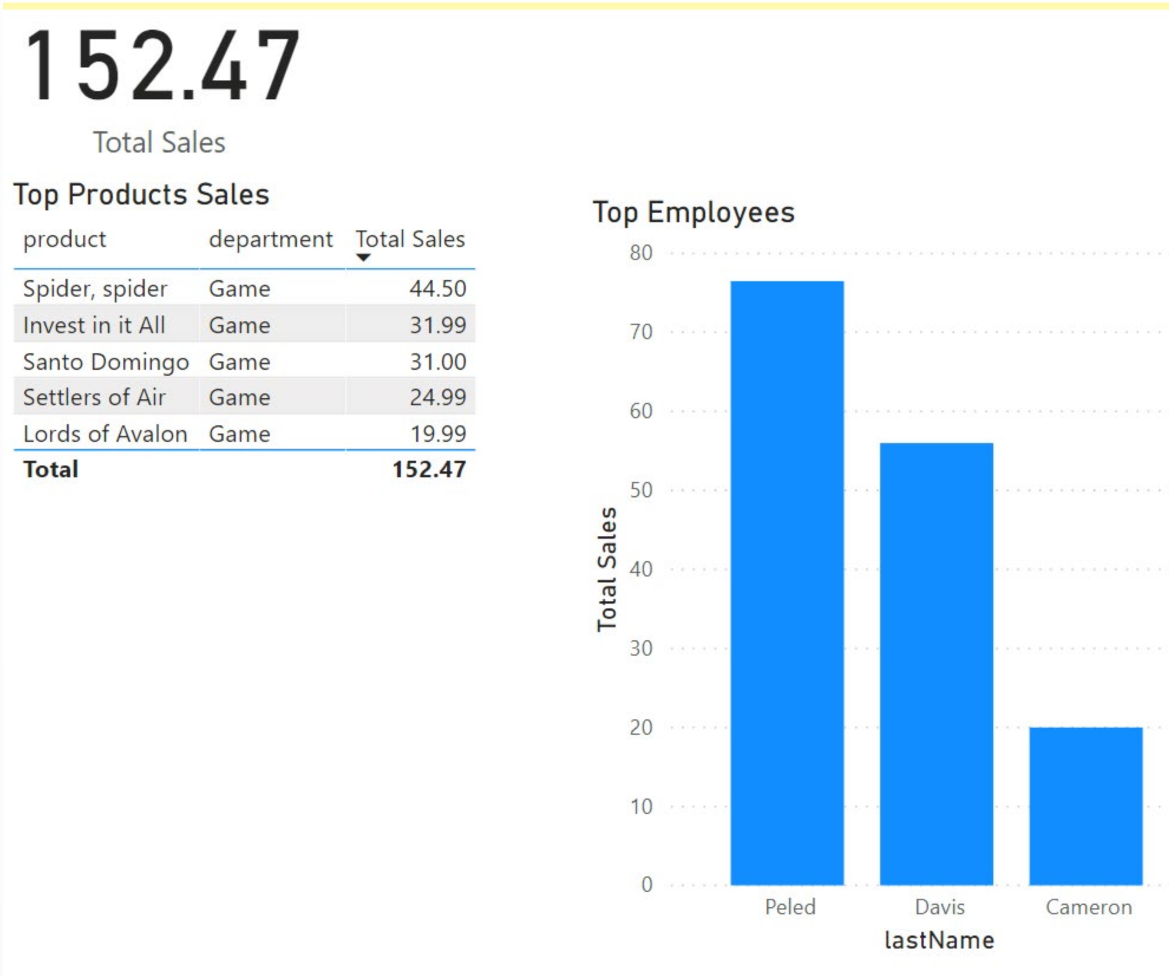
Notice how we apply the DAX filter on the dimension table. Row-level security performs better when the data is organized in a star schema. Apply the DAX filter to a dimension table, like we have done with the Products table here.

Remember that the DAX filter is applied to every interaction, slicer, and filter that the user uses. If we have a poor performing DAX filter, it will negatively impact the user experience. Keep the DAX filter as simple as possible.

Test the roles in Power BI Desktop

You can validate that this is working by selecting the **Modeling** tab, and select **View as Roles**.

The View as roles window appears and you can select the Game role. The report now renders as if you were in that role. You will only see the records that have a department of "Game."



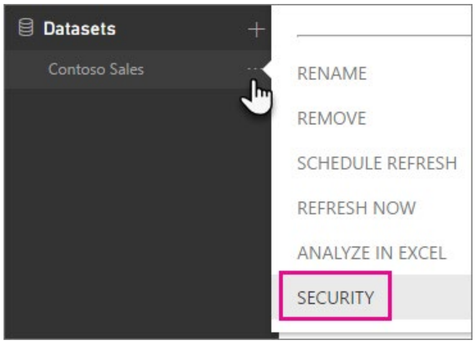
You can undo this by selecting view as roles again and selecting None.

Deploy the report to the Power BI service

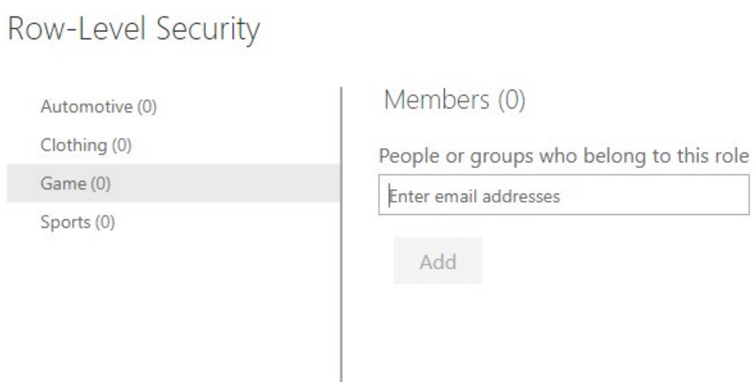
Deploy the report to the Power BI service by selected the **Publish** button on the **Home** tab. Choose a workspace.

Add members to the role in the Power BI service

Navigate to your workspace in the Power BI Service. Find the dataset you created with the same name as your report. Click the ellipsis button and select Security.

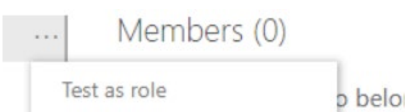


The Row-Level Security screen appears. From here, you can add active directory users and security groups to the security role. When members are added to this role, they will have the DAX filter that you previously defined applied to them. If members are not added to the role, but they have access to the report, RLS will not apply to them. You can add the three people in the Game department to the Game role. Now when those members log in, they will only see the report with data that applies to them.



Test the roles in the Power BI Service

You can also test this inside the Power BI Service. If you select the ellipsis next to the Game role on this screen, you can see Test as role.

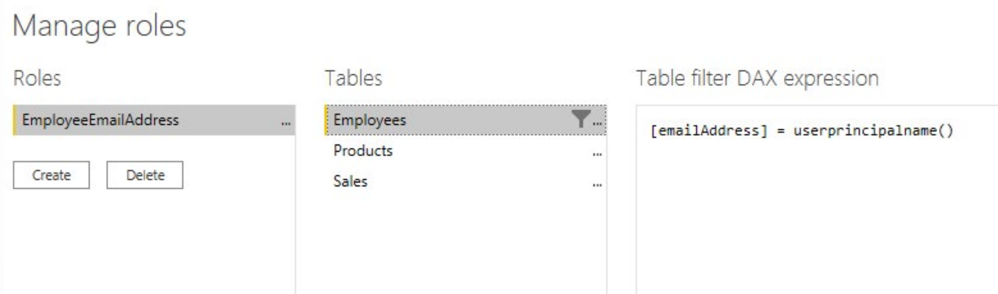


This will display the report as if you were a member of the role, in the Power BI service.

And there you have it! We've successfully implemented row-level security in Power BI.

Dynamic Method

There's a way to set up row-level security only once, without the need to continue to maintain it dynamically. We would like Power BI row-level security to only show sales to the user that they individually made. In our example, Russel King has made four sales. When he visits our report, he should only see the sales he's responsible for, and no other sales. We can configure row-level security exactly the way we configured it before, with only a single change. Instead of creating four roles, we only need one role. The DAX filter for that role would look like this:



Notice that instead of the fixed string we used before like "Game" or "Clothing", we're using a DAX function in the row-level security filter. This function, userprincipalname(), will compare the email address from the Employees table with the email that the user uses to log into the Power BI service. If he uses the email address `russel@tailwindtraders.com` to sign in to the Power BI service, it will compare that value to the email address in the Employees table. Assuming there's a relationship created between Employees and Sales, Russel will only see his four sales.

Module Review

In this module, we learned about row-level security, the ability in Power BI to limit what a user sees on a specific report. RLS targets the data to a specific user, for instance, only allowing a manager to see the salary of their direct reports. RLS is implemented with a combination of Power BI Desktop and the Power BI Service. To implement RLS, you create a DAX formula that restricts their data access. This makes RLS very versatile. You can use DAX to say that someone can only see records in the United States or sales transactions that are below a certain dollar amount. This programmatic approach means that RLS can be used in a wide-variety of solutions. Once the DAX formula is created in a specific security role, deploy the report and then add users to that role. RLS is an easily implemented, very powerful security feature of Power BI.

Knowledge Check

Question 1

Which function will tell you the logged on username in the Power BI Service?

☐ LOOKUPVALUE()

☐ USERPRINCIPALNAME()

☐ USEROBJECTID()

Question 2

Where can you test RLS by using different security roles?

☐ Power BI Desktop only

☐ Power BI Service only

☐ Both Power BI Desktop and Power BI Service

Answers

Question 1

Which function will tell you the logged on username in the Power BI Service?

☐ LOOKUPVALUE()

■ USERPRINCIPALNAME()

☐ USEROBJECTID()

Question 2

Where can you test RLS by using different security roles?

☐ Power BI Desktop only

☐ Power BI Service only

■ Both Power BI Desktop and Power BI Service