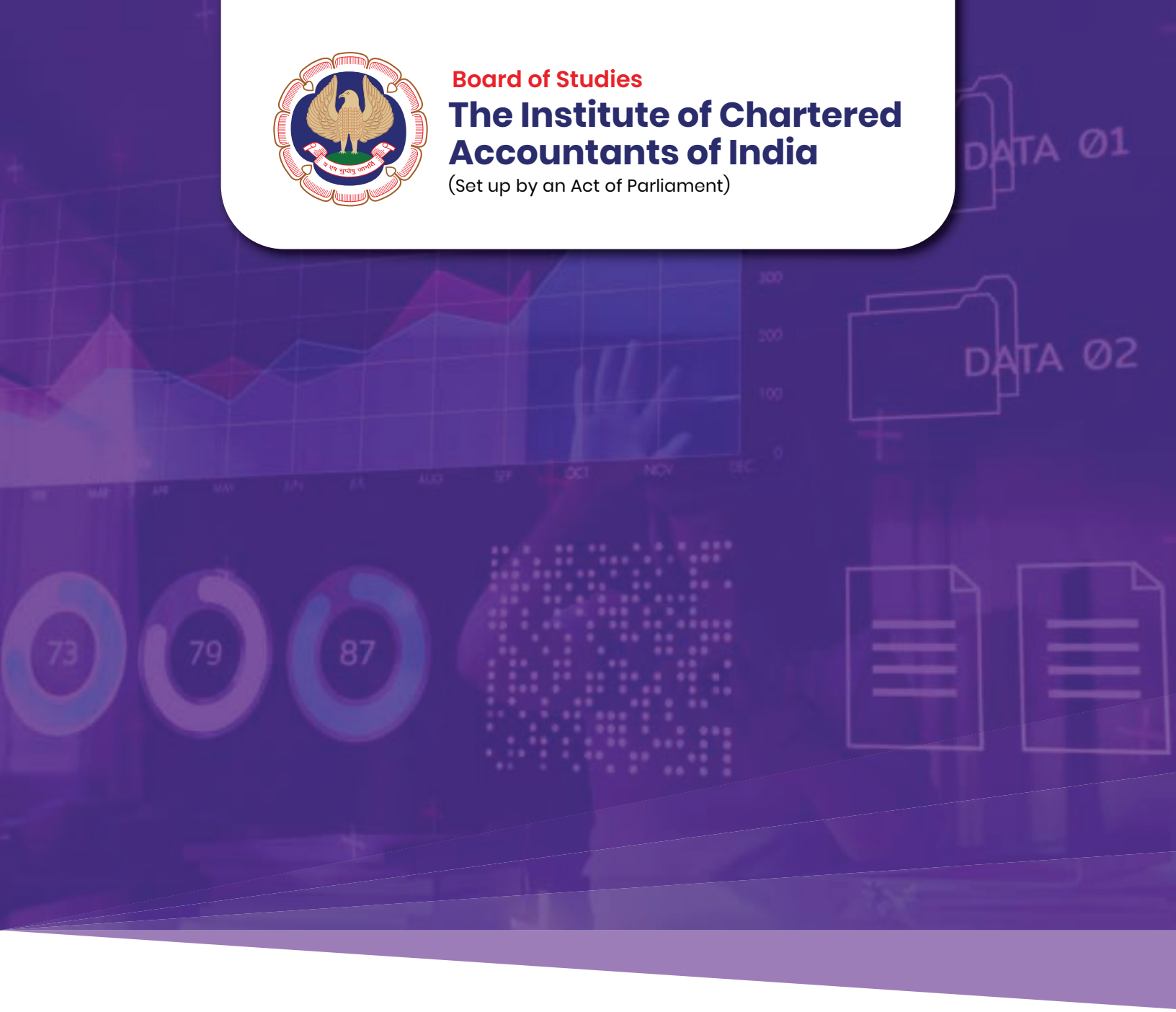
Board of Studies
The Institute of Chartered Accountants of India
(Set up by an Act of Parliament)



ADVANCED INTEGRATED COURSE ON

INFORMATION TECHNOLOGY AND SOFT SKILLS (AICITSS)

PART-A
ADVANCED INFORMATION TECHNOLOGY

MODULE 2

www.icai.org

य एष सुप्तेषु जागर्ति

ADVANCED INTEGRATED COURSE ON INFORMATION TECHNOLOGY (AICITSS)

COURSE MATERIAL MODULE - II

Board of Studies

The Institute of Chartered Accountants of India, New Delhi

Advanced Integrated Course on Information Technology and Soft Skills (AICITSS) Part-A (Advanced Information Technology)

The objective of the Study Material is to provide teaching material to the students to enable them to obtain knowledge in the subject. In case students need any clarifications or have any suggestions for further improvement of the material contained herein, they may write to the Joint Director, Board of Studies.

All care has been taken to provide interpretations and discussions in a manner useful for the students. However, the Study Material has not been specifically discussed by the Council of the Institute or any of its Committees and the views expressed herein may not be taken to necessarily represent the views of the Council or any of its Committees.

Permission of the Institute is essential for reproduction of any portion of this material.

Basic draft of this publication was prepared by CA. (Dr.) Rashmi Goel

Updated Edition	:	January, 2025
Committee/Department	:	Board of Studies
E-mail	:	helpdeskadvitt@icai.in
Website	:	www.icai.org
Sale Price	:	₹ 720/- (For All Modules)
ISBN	:	978-93-48313-58-4
Published by	:	The Publication & CDS Directorate on behalf of The Institute of Chartered Accountants of India ICAI Bhawan, Post Box No. 7100, Indraprastha Marg, New Delhi – 110 002 (India)
Printed by	:	Sahitya Bhawan Publications, Hospital Road, Agra – 282 003 January | 2025 | P3760 (Updated)

The Institute of Chartered Accountants of India
(Setup by an Act of Parliament)
Board of Studies

INDEX

य एष सुप्तेषु जागर्ति

UNIT

1

Microsoft Power BI

CHAPTER 1
INTRODUCTION TO POWER BI

LEARNING OBJECTIVES

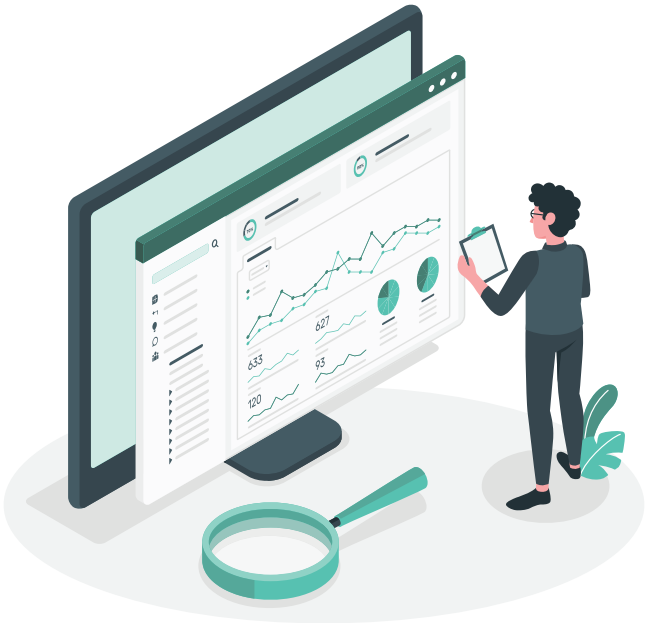
- Understand Power BI's role-specific benefits.
- Create a report and dashboard.
- Explore role-based features.
- Collaborate and share reports.
- See real-world role customization examples.

Begin exploring the fundamentals of Power BI interactive reports and the web-based Power BI service for your business.

1.1. DESCRIBE THE CAPABILITIES OF MICROSOFT POWER BI:

Today's business world is becoming increasingly more data-driven. Small and large businesses alike use data to make decisions about sales, hiring, goals, and all areas for which they have data. While most businesses have access to data of one type or another, it can be intimidating for your average business user to understand the data without a background in data analytics or statistics. Even for those individuals who do understand the data, it can be challenging to display it in an easily understood format.

Power BI takes the intimidation and hassle out of data analysis and visualization. By connecting to one or more of the hundreds of existing data sources through a secure and simple interface, you can quickly and interact with and understand your data to influence all business systems.



1.1.1. DESCRIBE USING POWER BI TO BUILD DATA-DRIVEN ANALYTICS:

From customer and employee data metrics for company goals to sales and acquisitions, businesses are drowning in data, but this data is only as good as your ability to interpret and communicate its meaning. That's where Power BI (Business Intelligence) comes into play.

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. From data in 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, clean, and model your data without affecting the underlying source, visualize (or discover) what's important, and share that with anyone or everyone you want.

Figure 1.1.1: How different sources of data can be showcased in one place.

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 phones and tablets.

Figure 1.1.2: Power BI Desktop, the Power BI service, and Power BI Mobile.

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

1.1.2. EXPLORE THE DIFFERENT POWER BI ELEMENTS:

To create Power BI Solutions, there are several major building blocks. These elements help not only what data is being presented, but also how it will appear to those who are consuming it. Those elements are datasets, reports, and dashboards. All elements are organized into workspaces, and they are created on capacities.Let's look at each of these elements in more detail.

Capacities:

Capacities are a core Power BI concept representing a set of resources used to host and deliver your Power BI content. Capacities are either shared or dedicated. A shared capacity is shared with other Microsoft customers, while a dedicated capacity is fully committed to a single customer. Dedicated capacities require a subscription. By default, workspaces are created in a shared capacity.

Workspaces:

Workspaces are containers for dashboards, reports, datasets, and dataflows in Power BI. There are two types of workspaces: My workspace and workspaces.
My workspace is the personal workspace for any Power BI customer to work with your own content. Only you have access to your My Workspace. You can share dashboards and reports from your My Workspace. If you want to collaborate on dashboards and reports or create an app, then you want to work in a workspace.

Workspaces are used to collaborate and share content with colleagues. You can add colleagues to your workspaces and collaborate on dashboards, reports, and datasets. With one exception, all workspace members need Power BI Pro licenses.

Workspaces are also the places where you create, publish, and manage apps for your organization. Think of workspaces as staging areas and containers for the content that will make up a Power BI app. So, what is an app? An app is a collection of dashboards and reports built to deliver key metrics to the Power BI consumers in your organization. Apps are interactive, but consumers cannot edit them. App consumers, colleagues who have access to the apps, do not necessarily need Pro licenses.

Datasets:

A dataset is a collection of data that you import or connect to. Power BI lets you connect to and import all sorts of datasets and bring all of it together in one place. Datasets can also source data from dataflows.

Datasets are associated with workspaces and a single dataset can be part of many workspaces. When you open a workspace, the associated datasets are listed under the Datasets tab. Each listed dataset represents a collection of data. For example, a dataset can contain data from an Excel workbook on OneDrive, an on-premises SSAS tabular dataset, and/or a Salesforce dataset. There are many different data sources supported. Datasets added by one workspace member are available to the other workspace members with an admin, member, or contributor role.

Shared Datasets:

Business intelligence is a collaborative activity. It's important to establish standardized datasets that can be the 'one source of truth.' Discovering and reusing those standardized datasets is key. When expert data modelers in your organization create and share optimized datasets, report creators can start with those datasets to build accurate reports. Your organization can have consistent data for making decisions and a healthy data culture. To consume these shared datasets just choose Power BI datasets when creating your Power BI report.

Reports:

A Power BI report is one or more pages of visualizations such as line charts, maps, and other elements. Reports can be created from scratch within Power BI, they can also be imported with dashboards that colleagues share with you, or Power BI can create them when you connect to datasets from Excel, Power BI Desktop, databases, and SaaS applications. For example, when you connect to an Excel workbook that contains Power View sheets, Power BI creates a report based on those sheets. And when you connect to a SaaS application, Power BI imports a pre-built report.

There are two modes to view and interact with reports:

- Reading view: When a report is opened by a user, it is displayed in the reading view.
- Editing view: For individuals who have edit permissions, the editing view is used to modify the different elements of the report and how they are presented.

When a workspace is opened, associated reports are listed under the Reports tab. Each listed report represents one or more pages of visualizations based on only one of the underlying datasets. To open a report, select it.

When you open an app, you are presented with a dashboard. To access an underlying report, select a dashboard tile (more on tiles later) that was pinned from a report. Keep in mind that not all tiles are pinned from reports, so you may have to click a few tiles to find a report.

By default, the report opens in Reading view. Just select Edit report to open it in Editing view (if you have the necessary permissions).

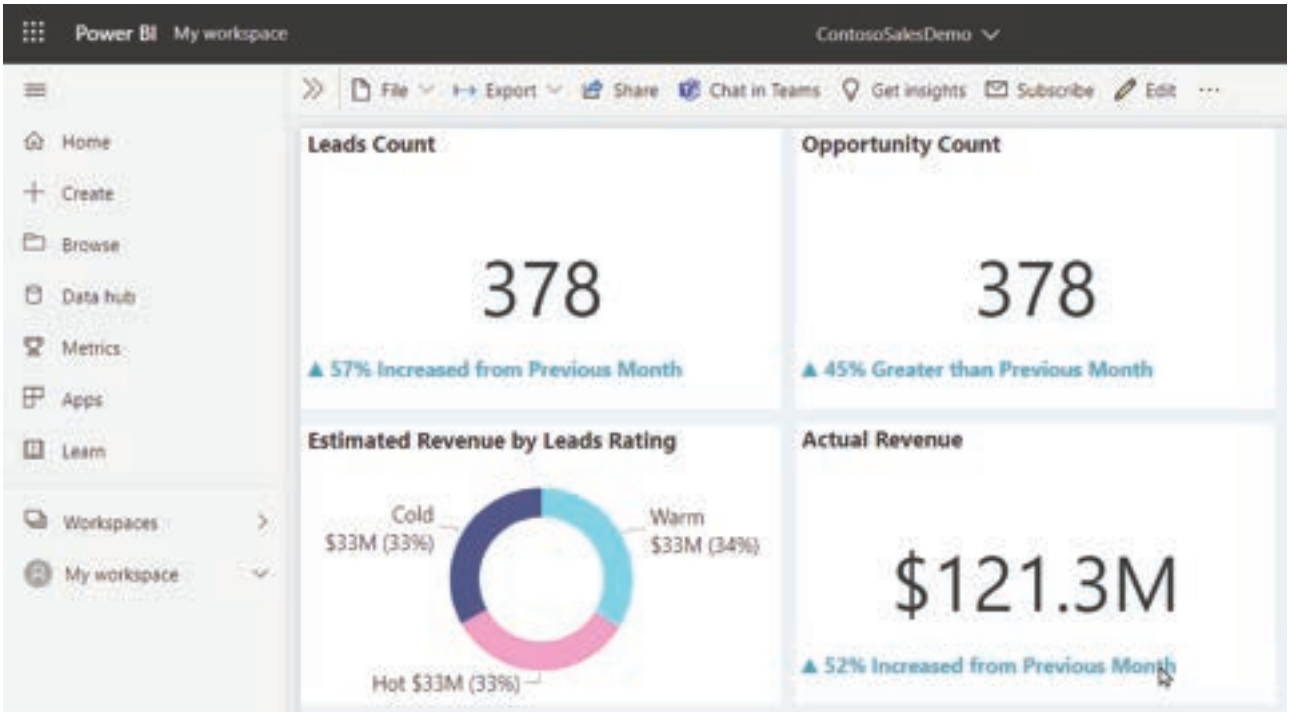


Figure 1.1.3.: Edit report button.

Dashboards:

A dashboard is a single canvas that contains zero or more tiles and widgets. Each tile pinned from a report or from Q&A displays a single visualization that was created from a dataset and pinned to the dashboard. Entire report pages can also be pinned to a dashboard as a single tile. There are many ways to add tiles to your dashboard, too many to be covered in this overview topic.

Why do people create dashboards? Here are just some of the reasons:

- To see all information needed to make decisions in one glance.
- To monitor the most important information about your business.
- To ensure all colleagues are on the same page, viewing and using the same information.
- To monitor the health of a business, product, business unit, or marketing campaign, etc.
- To create a personalized view of a larger dashboard and show all the metrics that matter to them.

When you open a workspace, the associated dashboards are listed under the Dashboards tab. To open a dashboard, select it. When you open an app, you will be presented with a dashboard. If you own the dashboard, you will also have edit access to the underlying dataset(s) and reports. If the dashboard was shared with you, you will be able to interact with the dashboard and any underlying reports but will not be able to save any changes.

Template Apps:

Power BI template apps enable Power BI partners to build Power BI apps with little or no coding and deploy them to any Power BI customer. As a Power BI partner, you create a set of out-of-the-box content for your customers and publish it yourself.
You can build template apps that allow your customers to connect within their own accounts. As domain experts, they can unlock the data in a way that is easy for their business users to consume.

Figure 1.1.4.: Power BI template apps.

1.1.3. DESCRIBE CLEANING AND TRANSFORMING DATA:

When working with data from different data sources, it's not always going to be in a format that allows it to be displayed with other data. For example, you might be getting ready to build a device usage report that displaysIoT-related device details for individual customers. Typically, that data exists in multiple systems. The customer details likely are in your organization's Customer Relationship Management (CRM) system, while the IoT data is likely stored in a dedicated IoT system such as Azure IoT Hub. Many times, the IoT data isn't structured as nicely as the data in your CRM system. Some of the data may not be in the right format, or there might simply be more data than you need. In these instances, you'll need to clean and transform your data. Cleaning and transforming data is how you prepare your data and get it ready to be used. To begin transforming and cleaning data, you use the Power BI Desktop application.

Power BI Desktop has three views:

- **Report view:** You can create queries to build compelling visualizations that you can share with others. You can arrange them as you want them to appear.

- **Data view:** See the data in your report in data model format, where you can add measures, create new columns, and manage relationships.

- **Model view:** Get a graphical representation of the relationships that are established in your data model and manage or modify them as needed.

Power BI Desktop includes the Power Query Editor tool, which can help you shape and transform data so that it's ready for your models and visualizations.

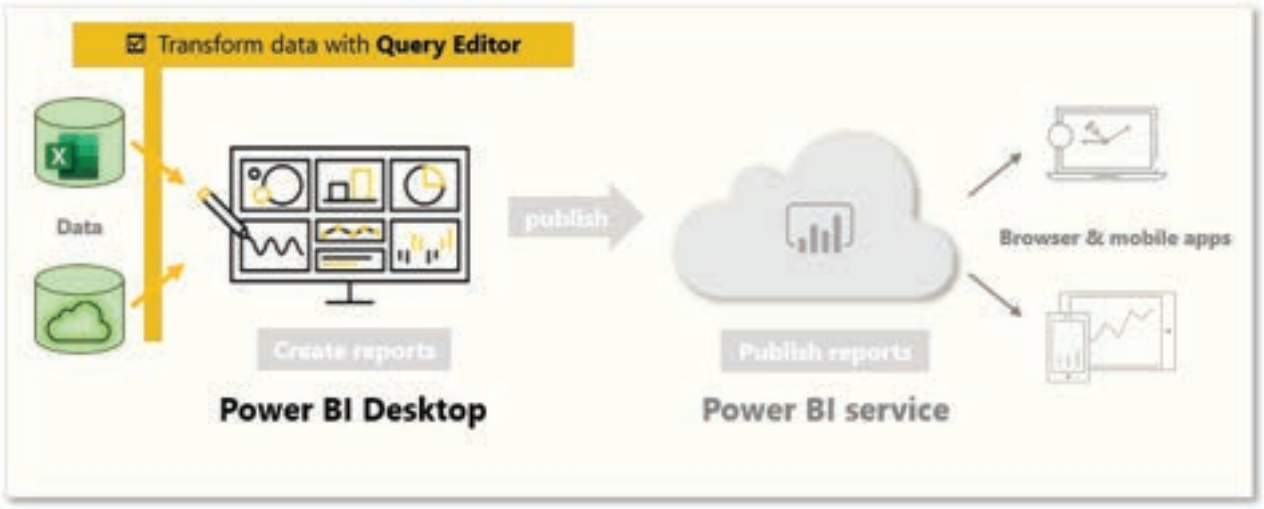


Figure 1.1.5.: Transform data with Query Editor.

To begin, select Edit from the Navigator window to launch Power Query Editor. You can also launch Power Query Editor directly from Power BI Desktop by using the Transform Data button on the "Home" ribbon.

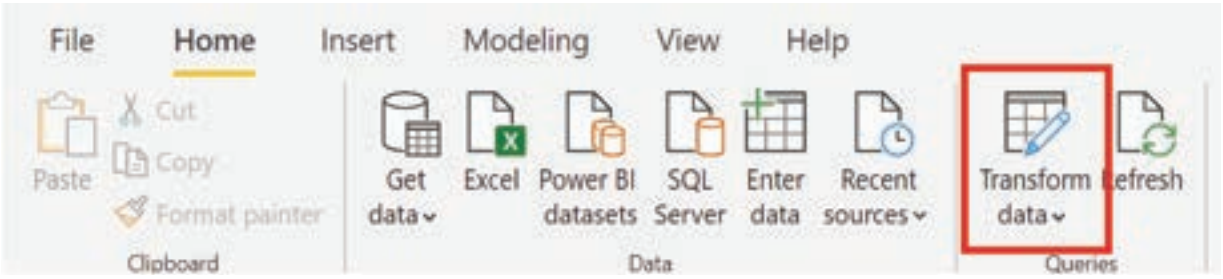


Figure 1.1.6.: Transform Data button in the Home tab.

Transforming data:

As mentioned previously, transforming data is the process of putting data into a format that is usable in your reports. Examples of the available transformations include removing a column from the table, duplicating the column under a new name, or replacing values.

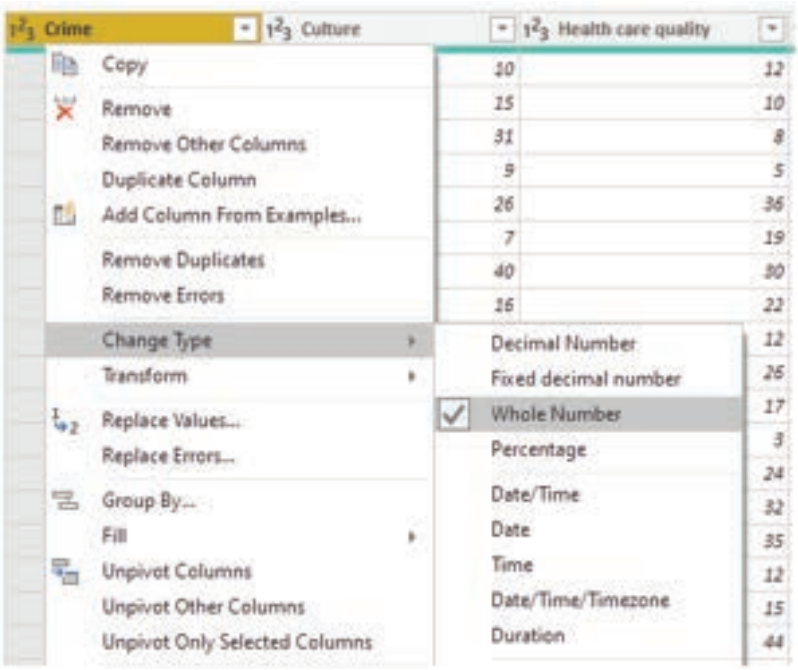
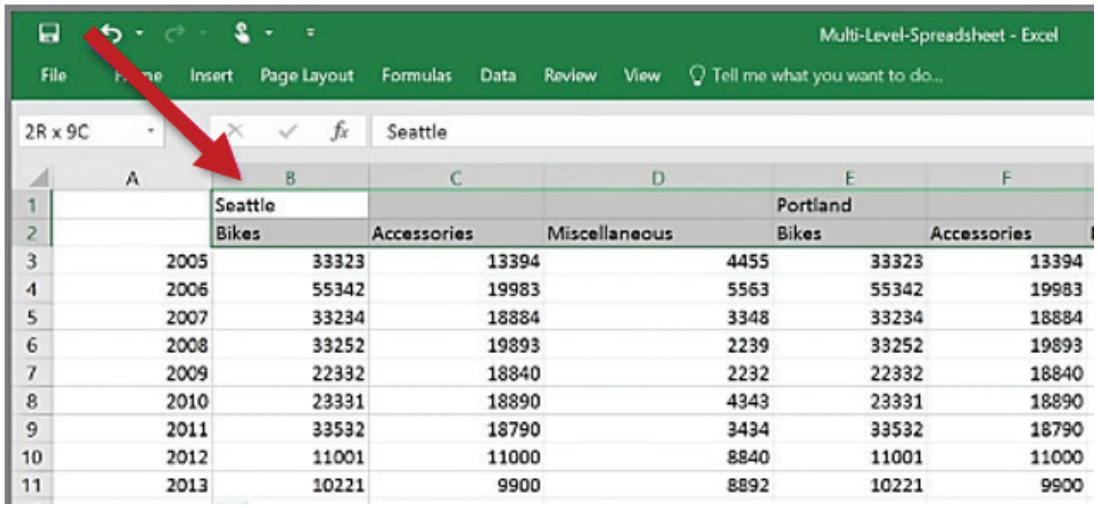


Figure 1.1.7.: Change Type menu.

Clean data:

While Power BI can import your data from almost any source, its visualization and modeling tools work best with columnar data. Sometimes, your data isn't formatted in simple columns, which is often the case with Excel spreadsheets.

A table layout that looks good to the human eye might not be optimal for automated queries. For example, the following spreadsheet has headers that span multiple columns.



	A	B	C	D	E	F
1		Seattle			Portland	
2		Bikes	Accessories	Miscellaneous	Bikes	Accessories
3	2005	33323	13394	4455	33323	13394
4	2006	55342	19983	5563	55342	19983
5	2007	33234	18884	3348	33234	18884
6	2008	33252	19893	2239	33252	19893
7	2009	22332	18840	2232	22332	18840
8	2010	23331	18890	4343	23331	18890
9	2011	33532	18790	3434	33532	18790
10	2012	11001	11000	8840	11001	11000
11	2013	10221	9900	8892	10221	9900

Figure 1.1.8.: Excel spreadsheet with headers that span multiple columns.

When you clean data, you might combine those rows into a single item to better format the data to fit your needs. Or you may have a series of numeric data that would need to be aggregated to display better. With Power Query, there are a series of tools that you can use to prepare the data.

1.1.4. BUILD A BASIC DASHBOARD:

Now that we have introduced you to some of the core concepts in Power BI such as reports, dashboards, and workspaces, let's see how these different elements would be used to support common business scenarios. Managers often want to see how their salespeople are performing. It would not be uncommon for your manager to ask to see your latest sales and profit figures by the end of the day. Many individuals keep those details in an Excel spreadsheet on their computer. Historically, getting that data into a format that a manager can easily consume could take several hours if not days.

With Power BI, we can easily create and share a dashboard with a manager by connecting to a data source such as an Excel spreadsheet on your laptop. While the data sources that you use might be different, the process for building and sharing a dashboard are same.

You need to take the following steps:

- Prepare your data: Preparing the data ensures that it's in a format that Power BI can easily consume.
- Build a report: The report contains the visuals that you want to include in your dashboard. Depending on the scenario, reports can be built in either Power BI Desktop or using the Power BI Service.
- Pin the report visuals to a dashboard: Dashboards are the primary element that users use for viewing data. They can include data from multiple reports as needed.
- Share a link to the dashboard: Any users with the link and the necessary permissions are easily able to view and interact with the data.

Prepare the data:

The first thing you need to do is ensure that your data is ready to be consumed. Depending on the data source and the volume of data you're working with, this might include doing some data cleansing and transforming using Power Query. In the case of connecting to an Excel spreadsheet, we want to make sure the data is in a flat table, and that each column contains the right data type. For example, text, date, number, or currency. It's also important that you have a header row but no columns or rows that display totals. Total operations will be handled in Power BI as we create the visuals.

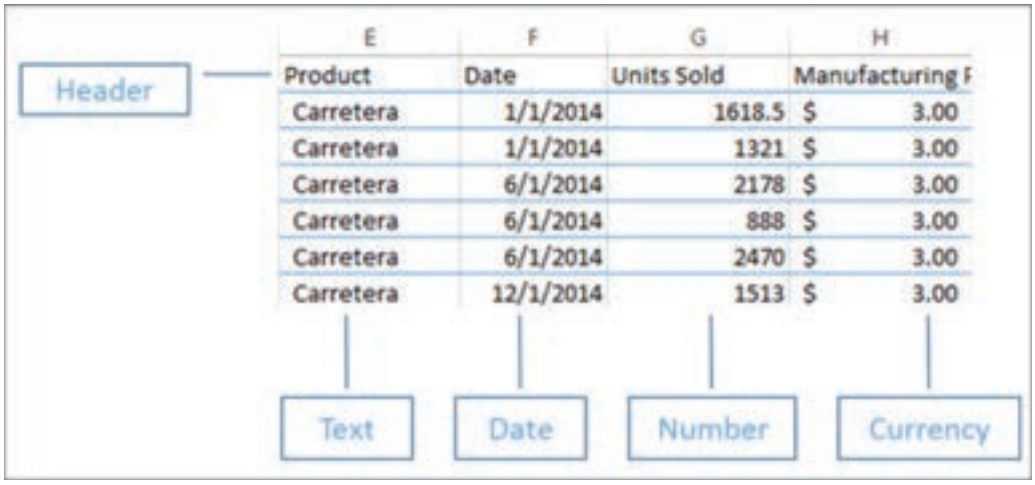


Figure 1.1.9.: Data organized in Excel.

You can see that there's a header column, and each column has the correct data type associated with the data.

Upload your data to the Power BI service:

The Power BI service is where you'll be able to create reports that connect to your data sources. This includes Excel files that live on your computer. With a few simple clicks, you can attach to a dataset, and Power BI will create a blank dashboard where you'll be able to place visuals later.

In the image, we attached to sample financial data. It shows the completed Financial Sample dataset, as well as the blank dashboard.

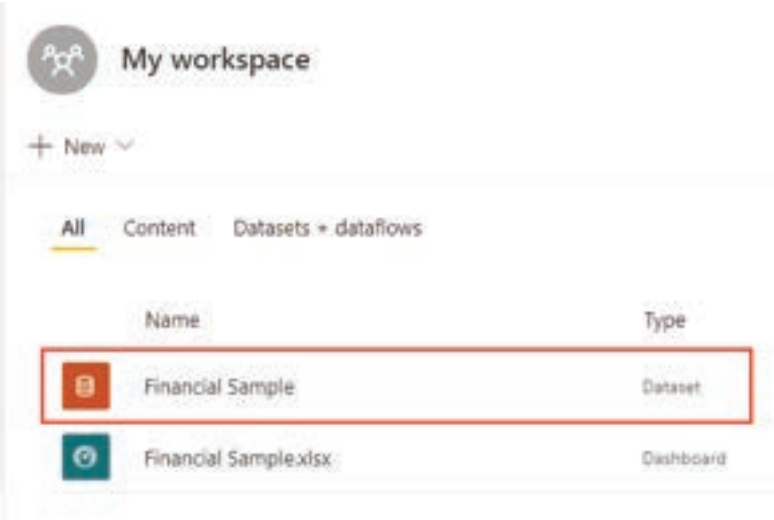


Figure 1.1.10.: My Workspace, highlighting the Financial Sample dataset.

Build your report:

Once you connect to your data, you can either create a new report or edit an existing report that was created previously. Then we can begin editing the report by using the editing view. On the right are the Visualizations, Filters, and Fields panes. Your Excel workbook table data appears in the Fields pane. At the top is the name of the table, financials. Under that, Power BI lists the column headings as individual fields.

Different visualizations are available that you can use to display the data in your report. Different visualizations can be added to the report based on the data that you want to communicate. In the image, multiple bar charts are being used to showcase the number of units being sold. Additionally, map control is included to show sales volumes per country/region.

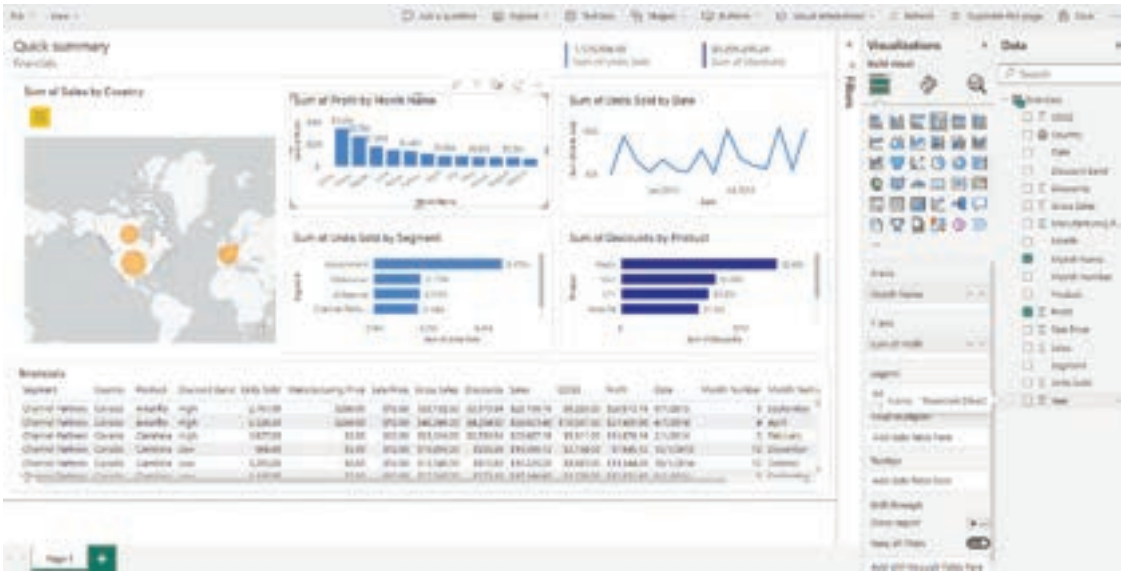


Figure 1.1.11.: Power BI report including charts and data tables.

Each visualization includes a series of filters and controls that can be used to impact how the data is being presented. For example, if you want to change the sum of units sold to display profit by date, you could switch the Y-axis from Sum of Units sold, to Sum of Profit.

More and more visualizations are being added to Power BI to reflect how business is being done. For example, there's a Power Apps visualization that lets you build a canvas app directly in your Power BI report, which is connected to the Power BI data set. As people interact with the data, the data in the Power App changes. For example, you could create a canvas app that includes actions such as sending emails or scheduling meetings. As you drill down on a Power BI report, you identify customers we have not talked to in a while. The embedded canvas app lets us initiate actions, such as sending an email, right from the Power BI report.

Pin to a dashboard:

After you have all your visualizations on your reports, you can build your dashboard. Dashboards are easy to build because you are just determining which visuals from your created reports you want to include.

In the image, we are taking the Profit by Date visualization and pinning it to a dashboard.

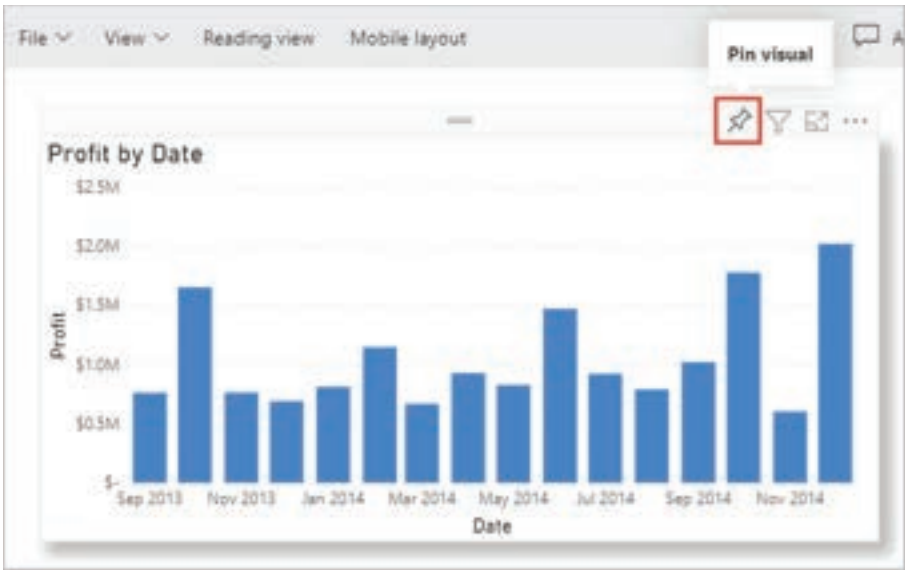


Figure 1.1.12.: A visual, highlighting the Pin visual icon.

Since dashboards can have visuals from multiple reports, it makes it easy to have very detailed dashboards that include data from multiple data sources, even if some of those data sources might be unrelated.

Figure 1.1.13.: A dashboard including data visualizations from multiple reports.

Share a link to your dashboard:

Initially dashboards that you create will only be visible to the person that created them. However, Power BI makes it simple to share dashboards with other individuals. You can share your dashboard and underlying report with any colleague who has a Power BI account. They'll be able to interact with it, but they cannot save changes. If you allow it, they can reshare with others or build a new report based on the underlying dataset.

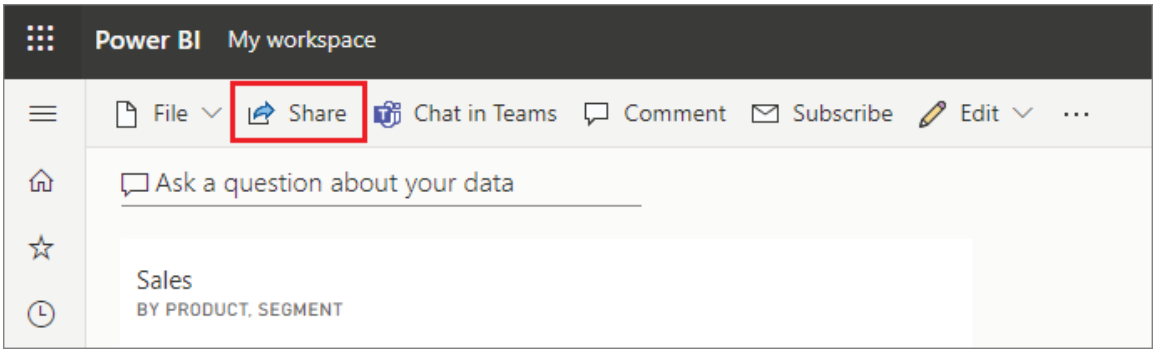


Figure 1.1.14.: My workspace, highlighting the Share icons.

SUMMARY

Microsoft Power BI is a robust business intelligence tool that offers the following capabilities:

Data Analysis and Visualization: Power BI allows users to connect to various data sources, analyze data, and create interactive visualizations to inform decision-making.

Key Elements: It comprises Power BI Desktop (for report creation), Power BI Service (cloud-based service), and Power BI Mobile (mobile apps). These elements enable users to create, share, and consume insights effectively.

Building Blocks: Power BI includes concepts like capacities (resource hosting), workspaces (containers for content), datasets (data collections), shared datasets (standardized data sources), reports (visualization pages), dashboards (interactive canvases), and template apps (pre-built apps).

Data Cleaning and Transformation: Power BI offers Power Query Editor to clean and shape data from various sources, ensuring it's suitable for analysis.

AI Insights: Power BI employs artificial intelligence to detect trends and anomalies, enhancing data analysis. This includes identifying anomalies, trends in time-series data, and KPI analysis.

Dashboard Creation: Creating a basic dashboard involves data preparation, report building with visualizations, pinning visuals to a dashboard, and sharing the dashboard with collaborators.

Business Value: Organizations like the Miami Heat basketball team have used Power BI to boost sales, reduce operational costs, and make accurate attendance predictions, resulting in improved business outcomes.

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 What is Power BI primarily used for?
   a) Video Editing
   b) Data Analysis and Visualization
   c) Web Development
   d) Graphic Design

2 Which of the following are the main components of Power BI?
   a) Word, Excel, PowerPoint
   b) Power BI Desktop, Power BI Service, Power BI Mobile
   c) Windows, macOS, Linux
   d) Internet Explorer, Chrome, Firefox

3 What is the purpose of Power Query Editor in Power BI?
   a) Creating reports
   b) Data cleansing and transformation
   c) Sharing dashboards
   d) Building datasets

4 Which Power BI component is used for creating reports and visualizations?
   a) Power BI Service
   b) Power BI Mobile
   c) Power BI Desktop
   d) Power BI Viewer

5 What are the building blocks of Power BI solutions?
   a) Datasets, Dashboards, Templates
   b) Workspaces, Reports, Tables
   c) Capacities, Workspaces, Reports
   d) Datasets, Reports, Dashboards

6 In Power BI, what is the difference between shared and dedicated capacities?
   a) Shared capacities are free, and dedicated capacities require a subscription.
   b) Shared capacities are fully committed to a single customer, and dedicated capacities are shared.
   c) Shared capacities are shared with other customers, while dedicated capacities are fully committed to a single customer.
   d) Shared capacities are used for workspaces, and dedicated capacities are used for datasets.

7 What is the primary purpose of a dashboard in Power BI?
   a) Data cleansing
   b) Data modeling
   c) Data visualization
   d) Data transformation

8 What is a template app in Power BI used for?
   a) Creating reports from scratch
   b) Sharing dashboards with external users
   c) Building canvas apps
   d) Publishing pre-built content for customers

9 Which view in Power BI Desktop allows you to add visualizations and arrange them for reports?
   a) Data view
   b) Report view
   c) Model view
   d) Query view

10 What is the purpose of Power BI's AI Insights feature?
   a) To format data for reporting
   b) To identify trends and anomalies in data
   c) To create data visualizations
   d) To share reports with colleagues

11 Which Power BI component is used to connect to and import data from various sources?
   a) Power BI Desktop
   b) Power BI Service
   c) Power BI Mobile
   d) Power Query Editor

12 How can you share a Power BI report with a colleague who has a Power BI account?
   a) By sending them an email attachment
   b) By exporting the report to PDF
   c) By sharing a link to the report in Power BI Service
   d) By saving the report as an Excel file

13 In Power BI, what does "pinning visuals to a dashboard" mean?
- a) Attaching physical charts to your monitor
- b) Creating visualizations
- c) Embedding reports into PowerPoint
- d) Adding report visualizations to a dashboard for quick access

14 What is the main difference between Reading view and Editing view in Power BI reports?
- a) Reading view allows only data viewing, while Editing view allows data editing.
- b) Reading view displays visuals, while Editing view is for data transformation.
- c) Reading view is for mobile devices, while Editing view is for desktop.
- d) Reading view is for creating reports, while Editing view is for consuming reports.

15 What is the benefit of using Power BI workspaces?
- a) They are personal spaces for data storage.
- b) They allow collaboration and sharing of content.
- c) They are required for Power BI Desktop.
- d) They are used for creating templates.

Answers

1 *b)*
2 *b)*
3 *b)*
4 *c)*
5 *d)*
6 *c)*
7 *c)*
8 *d)*
9 *b)*
10 *b)*
11 *a)*
12 *c)*
13 *d)*
14 *a)*
15 *b)*

SELF-EXAMINATION QUESTIONS FOR PRACTICE:

1 What are the main components of Microsoft Power BI, and how do they contribute to data analysis and visualization?

2 Explain the difference between shared and dedicated capacities in Power BI, and when would you use each?

3 How does Power BI's Power Query Editor assist in data cleaning and transformation? Can you provide an example of a data transformation task?

4 What is the significance of AI Insights in Power BI, and how can they help users identify trends and anomalies in their data?

5 How does Power BI promote collaboration among users when working on reports and dashboards within an organization?

6 Describe a real-world business scenario where Power BI has demonstrated its value in data analysis and decision-making?

CHAPTER 1

1.2. GET STARTED BUILDING WITH POWER BI

LEARNING OBJECTIVES

- After completing this module on "How Power BI matches your role," learners will be able to understand the key components of Power BI, including Power BI Desktop, the Power BI service, and Power BI Mobile apps, and how they are used in various roles and workflows within an organization. Additionally, learners will gain proficiency in creating datasets, reports, visualizations, and dashboards, allowing them to effectively transform data into insightful reports and share them through the Power BI service and mobile apps.

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.

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.

1.2.1. 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:

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

Figure 1.2.1.: Power BI cycle of use

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.

1.2.2. BUILDING BLOCKS OF POWER BI:

In Microsoft Power BI, there are basic building blocks that make up the reports and dashboards consumed by end users. Think of it similarly to the basic construction materials that can be used to build homes or other structures.

Here are the basic building blocks of Power BI:

- Reports consist of visualizations and datasets created with the Power BI Desktop application.
- Dashboards consist of tiles from report visualizations created in the online Power BI service.

Visualizations:

A visualization (or 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 Power BI.

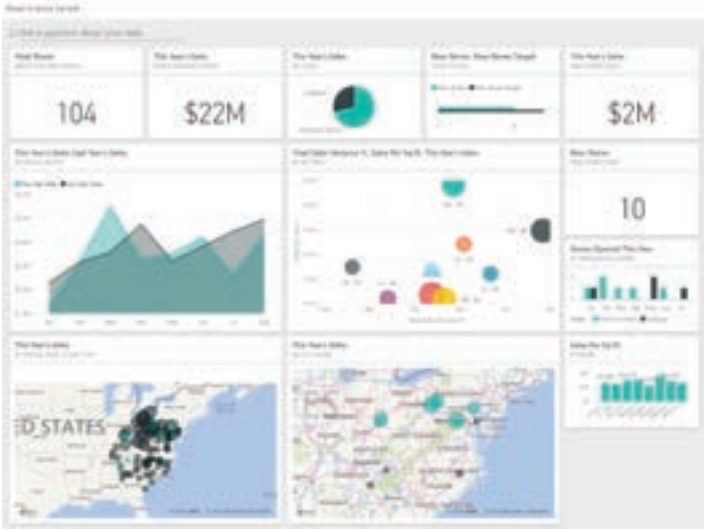


Figure 1.2.2.: Sample Power BI dashboard with various visualizations.

Visualizations can be simple, like a single number that represents something significant. Visuals can also be 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.

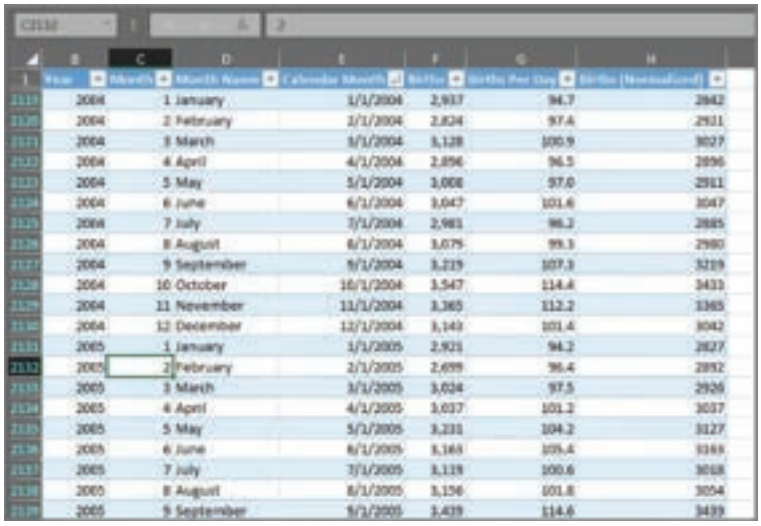


Figure 1.2.3.: Excel file with contents from a data table.

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.

You can create a Power BI report from almost any data, thanks to the many available data

connectors, such as Excel, a Microsoft SQL Server database, Azure, Oracle, Facebook, Salesforce, and MailChimp.

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 on one or more pages. As with other reports you've seen or created, Power BI reports combine related data. The following image shows a report in Power BI Desktop—in this case, it's the second page in a five-page report.

Figure 1.2.4.: Report in Power BI Desktop

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 report or a collection of visualizations, you can create a Power BI dashboard. Much like the dashboard in a car, a dashboard is a selected group of visuals that provide quick and important insight into the data or story you're trying to present.

Dashboards are limited to a single page and allow users to follow a visual to the underlying report. Users interact with dashboards through the Power BI service or on a mobile device.

Tiles:

In Power BI, a tile is a single visualization on 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.

Figure 1.2.5.: Power BI dashboard with tiles and a single tile highlighted.

When you're creating 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:

Let's review the building blocks of Power BI:

- Power BI Desktop lets you build datasets and use visuals to make reports.
- The online Power BI service brings together reports, dashboards, and tools for easy distribution and management of your Power BI content.

Understanding the Power BI basics empowers you to create datasets and design reports. Your reports don't have to be complex to be interesting and informative. Power BI offers easy ways to design reports from a single Excel sheet.

Power BI is also scalable, allowing you to create datasets from various data sources, even incorporating custom code. The dataset can then be used to design interactive reports and dashboards that emphasize crucial data for informed business decisions.

No matter how you use Power BI, it all starts with datasets and visuals. These are the foundation for your reports that share insights and dashboards that present the most important data upfront.

1.2.3. TOUR AND USE THE POWER BI SERVICE:

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 distribute to consumers to view through the service or mobile app.

Power BI service allows you to create apps for easy distribution and clutter-free consumption. An app is a way to group related reports and dashboards and distribute to the appropriate audience(s).

We go into more detail about apps (and the service) in upcoming modules, but let's walk through the experience to understand how apps benefit your organization.

Explore built-in sample reports:

Easily explore built-in samples to get familiar with using the Power BI service. Built-in samples are each a bundle of one or more dashboards, datasets, and reports that you can use with the Power BI service.

From the Power BI service, open the Learning Center from the left navigation pane. Pick one of the available built-in samples, which opens in Reading mode. Power BI then imports the sample and adds a new report and dataset to your My workspace.

Figure 1.2.1.: Power BI cycle of use

After you've chosen a sample report, you can see the direct report-sharing experience for consumers. Take note of the Power BI service navigation pane and header, as shown in the following screenshot. You can see the report navigation and the filter pane, both of which are collapsible.

Figure 1.2.7.: Screenshot of the built-in Human Resources sample report.

Explore template apps:

Now that you understand how a report can be shared through the Power BI service, let's look at the app experience. To replicate the experience, we're using the GitHub template app.

Figure 1.2.8.: Power BI Template apps.

Tip:

To access template apps, select the Apps icon from the left navigation pane > Get apps > Template apps.

In the following screenshot, you can see that the Power BI service left navigation pane and

header that were visible with the direct report are gone. Also, note the dashboard and a multi-page report in the app navigation pane. The app provides a cleaner look with only the relevant content. It's also customizable with app color and thumbnail. Apps also allow you to configure multiple audiences if you need to limit access to certain pages in a report, for instance.

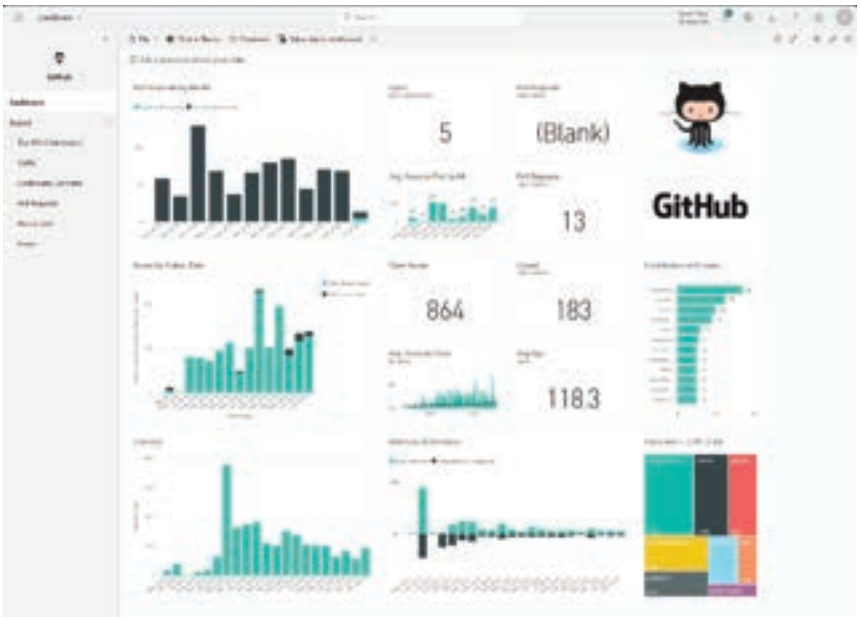


Figure 1.2.9.: Github app.

All of the visuals are interactive and interacting with one visual filters the others accordingly. For example, when you select on mihart in the donut chart on the Top 100 Contributors report, all other visuals only show related data for mihart.

Figure 1.2.10.: Filtered Top 100 Contributors report page.

Refresh data in the Power BI service:

Likely, your data changes regularly, so Power BI accounts allows on-demand and scheduled dataset refreshes. From the app workspace, you manually refresh or schedule up to eight refreshes per day at minimum.

Tip:

For more information about all refresh schedules, see the Refresh data documentation.

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.

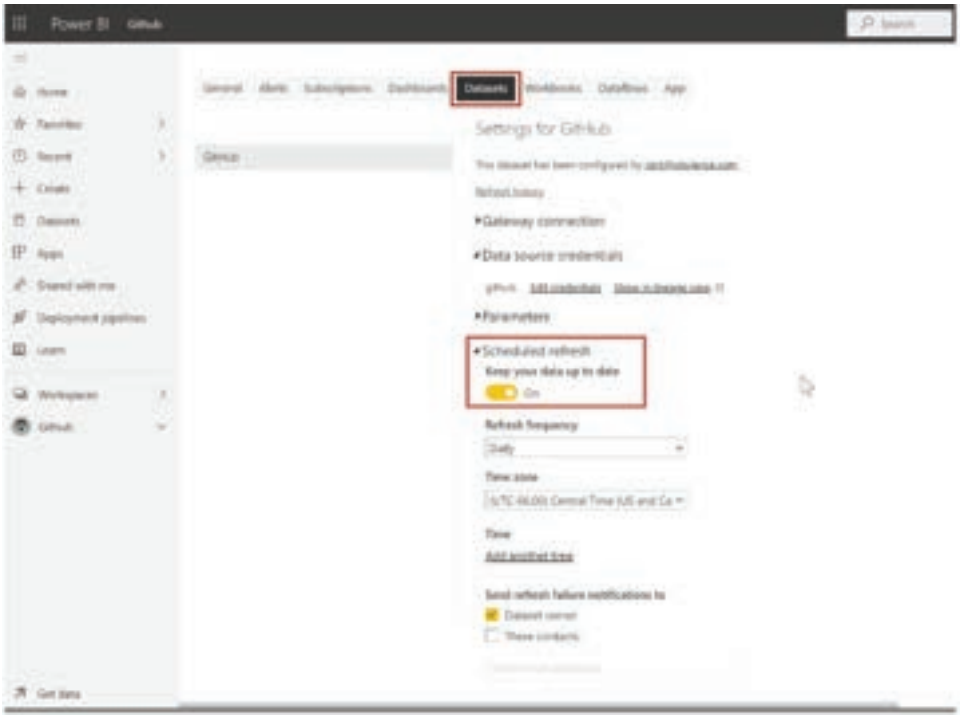


Figure 1.2.11.: Datasets setting tab with Scheduled refresh section highlighted

SUMMARY

- Power BI is a comprehensive suite of software services, apps, and connectors designed to transform diverse data sources into coherent and visually appealing insights. It consists of Power BI Desktop (a Windows application), the Power BI service (a SaaS offering), and mobile Power BI apps for various devices.

- The usage of Power BI can vary based on one's role, with different elements catering to different needs. Power BI Desktop is used for report creation, while the Power BI service allows for online visualization and dashboard building. Mobile apps enable users to access and interact with reports and dashboards on the go.

- The key building blocks of Power BI are reports, dashboards, visualizations, and datasets. Reports comprise visualizations and datasets created using Power BI Desktop. Visualizations are graphical representations of data, while datasets can be simple or complex collections of data from various sources. Dashboards are a selection of visuals that provide quick insights and are shared via the Power BI service.

- The content emphasizes the importance of datasets and visuals as the foundation for creating informative reports and dashboards in Power BI. It also introduces the flow of work in Power BI, starting from report creation in Power BI Desktop to publishing and sharing via the Power BI service.

- The Power BI service is further explored, including the use of apps for distributing reports and dashboards to specific audiences. Built-in sample reports and template apps are introduced, along with the ability to refresh data in the Power BI service to keep reports up-to-date.

- The Power BI service provides a simple and interactive user experience to take your data analytics to the next level.

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 What are the three main elements of Power BI?
- a) Reports, Dashboards, and Visualizations
- b) Power BI Desktop, Excel, and Service
- c) Desktop, Web, and Mobile
- d) Datasets, Filters, and Charts

2 Which tool is primarily used to create datasets and reports in Power BI?
- a) Power BI Mobile app
- b) Excel
- c) Power BI Desktop
- d) Power BI Service

3 What is the purpose of Power BI Mobile apps?
- a) To create datasets
- b) To design reports
- c) To view and interact with shared dashboards and reports on mobile devices
- d) To publish reports to the Power BI service

4 What is a visualization in Power BI?
- a) A dataset
- b) A single number
- c) A visual representation of data, like a chart or map
- d) A collection of filters

5 What is a dataset in Power BI?
- a) A single number
- b) A collection of filters
- c) A collection of data used to create visualizations
- d) A Power BI report

6 What is the primary purpose of a Power BI report?
- a) To create datasets
- b) To organize visualizations onto one or more pages
- c) To filter data
- d) To view data on a mobile device

7 What is a Power BI dashboard?
- a) A collection of filters
- b) A single number
- c) A selected group of visuals that provide quick insights
- d) A dataset

8 What are tiles in Power BI?
a) A type of visualization
b) The rectangular boxes that hold individual visuals on a dashboard
c) A dataset
d) A type of filter

9 What is the common flow of work in Power BI, starting with Power BI Desktop?
a) Create a dataset, then view dashboards
b) Create a dataset, publish to Power BI Service, and share dashboards
c) View dashboards, then create a dataset
d) Share dashboards, then create a dataset

10 What is the purpose of Power BI apps?
a) To create datasets
b) To group related reports and dashboards for distribution
c) To design visualizations
d) To filter data

11 How can you refresh data in the Power BI service?
a) By manually refreshing it from the app workspace
b) By scheduling refreshes at any time
c) By scheduling up to eight refreshes per day
d) Data cannot be refreshed in the Power BI service

12 Which component of Power BI is used for creating visualizations and reports?
a) Power BI Service
b) Power BI Mobile app
c) Power BI Desktop
d) Power BI Dashboard

13 What is the primary purpose of a Power BI report?
a) To create datasets
b) To organize visualizations
c) To schedule data refreshes
d) To group related reports

14 What is the primary role of Power BI Mobile apps?
a) To create datasets
b) To design reports
c) To view and interact with shared dashboards and reports on mobile devices
d) To publish reports to the Power BI service

15 Which Power BI element is a selected group of visuals that provide quick insights into data?
a) Dataset
b) Report
c) Dashboard
d) Tile

Answers

1 *c) Desktop, Web, and Mobile*
2 *c) Power BI Desktop*
3 *c) To view and interact with shared dashboards and reports on mobile devices*
4 *c) A visual representation of data, like a chart or map*
5 *c) A collection of data used to create visualizations*
6 *b) To organize visualizations onto one or more pages*
7 *c) A selected group of visuals that provide quick insights*
8 *b) The rectangular boxes that hold individual visuals on a dashboard*
9 *b) Create a dataset, publish to Power BI Service, and share dashboards*
10 *b) To group related reports and dashboards for distribution*
11 *c) By scheduling up to eight refreshes per day*
12 *c) Power BI Desktop*
13 *b) To organize visualizations*
14 *c) To view and interact with shared dashboards and reports on mobile devices*
15 *c) Dashboard*

SELF-EXAMINATION QUESTIONS FOR PRACTICE:

1 What is Power BI, and how does it help turn disparate data sources into actionable insights?

2 What are the three main elements that make up Microsoft Power BI, and how do they work together?

3 How might your usage of Power BI differ from that of a colleague in a different role?

4 What is the common flow of work in Power BI, from data ingestion to consumption?

5 What are visualizations in Power BI, and why are they important in data analysis?

6 How do reports differ from dashboards in Power BI, and what role do they play in presenting data?

7 What is the significance of tiles in Power BI dashboards, and how can they be customized?

CHAPTER 2
CONNECTING TO DATA SOURCES WITH POWER BI DESKTOP:

How can you find, collect, and clean data from different sources? Power BI is a tool for making sense of your data. You will learn tricks to make data-gathering easier.

2.1. GET DATA WITH POWER BI DESKTOP

LEARNING OBJECTIVES

- Explore the data-centric features and tools of Power BI.
- Explore ways to find data.

2.1.1. OVERVIEW OF POWER BI DESKTOP:

Power BI Desktop is a free application for PCs that lets you gather, transform, and visualize your data. In this module, you'll learn how to find and collect data from different sources and how to clean or transform it. You'll also learn tricks to make data-gathering easier.

Power BI Desktop and the Power BI Service work together. You can create your reports and dashboards in Power BI Desktop, and then publish them to the Power BI Service for others to consume.

The following are the tasks that you will complete in this module:

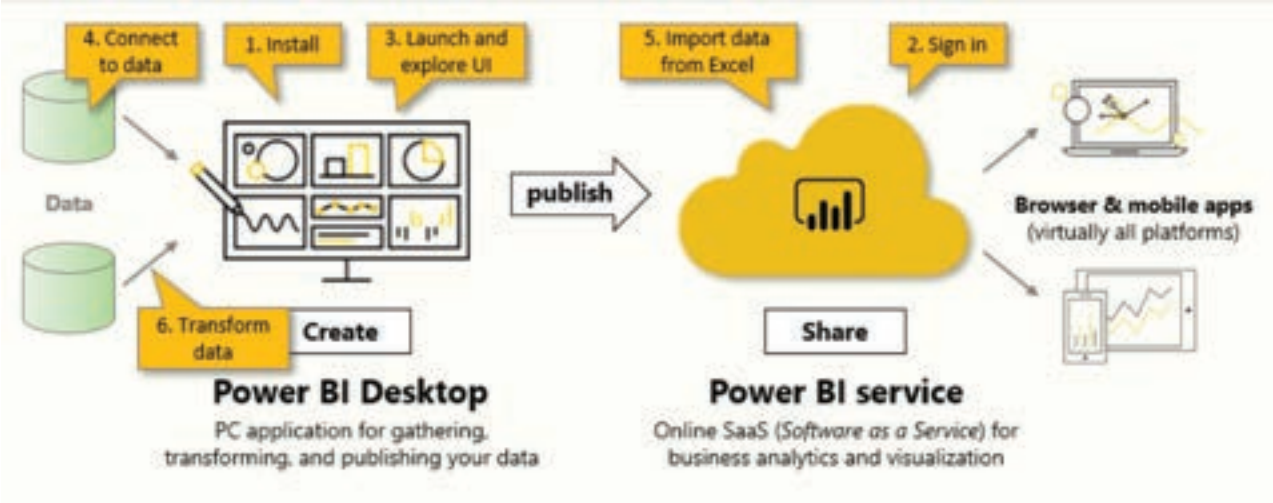


Figure 2.1.1.: Installing, signing-in, connecting to data, and transforming data.

To perform the exercises in this module, you'll need to have Power BI desktop installed and have a Power BI Service account set up.

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.

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 and sign up with your email address.

Download sample data:

To follow along with the examples in the videos and on the pages, download the sample Excel workbook and import into Power BI Desktop (Get Data > Excel).

2.1.2. EXPLORE POWER BI DESKTOP

The idea of building and sharing reports is an abstract concept. It will make more sense if you explore Power BI Desktop hands-on. The first step is to launch and explore the user interface (UI).

In this unit, you will:

Launch the Power BI Desktop.
Explore the UI.

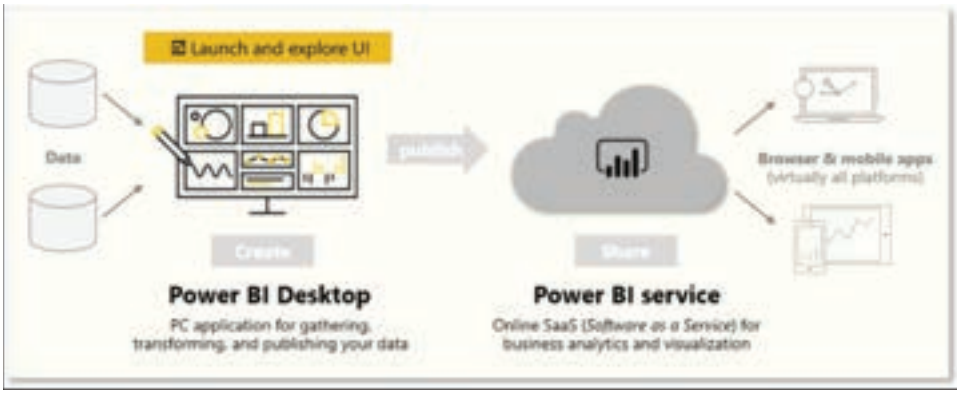


Figure 2.1.2.: Launch and explore the Power BI UI

Note:

To follow along with the examples in this module, download the sample Excel workbook here and import into Power BI Desktop (Get Data > Excel) if you haven't already.

Launch Power BI Desktop:

When you launch Power BI Desktop, the Getting Started dialog box will appear, which provides useful links to forums, blogs, and introductory videos. Close this dialog box for now, but keep the Show this screen on startup option selected so that you can explore it later.

Explore the report building environment:

In Power BI Desktop, you'll begin to build reports in the Report view. You'll be working in five main areas:

Figure 2.1.3.: The five areas of Report view.

- Ribbon - Displays common tasks that are associated with reports and visualizations.
- Report view, or canvas - Where visualizations are created and arranged. You can switch between Report, Data, and Model views by selecting the icons in the left column.
- Pages tab - Located along the bottom of the page, this area is where you would select or add a report page.
- Visualizations pane - Where you can change visualizations, customize colors or axes, apply filters, drag fields, and more.
- Fields pane - Where query elements and filters can be dragged onto the Report view or dragged to the Filters area of the Visualizations pane.

Tip:

You can collapse the Visualizations and Fields panes to provide more space in the Report view by selecting the small arrow, as shown in the following screenshot.

Figure 2.1.4.: Collapse or expand the Visualizations and Fields.

Create a visual:

To create a visual, drag a field from the Fields list onto the Report view.

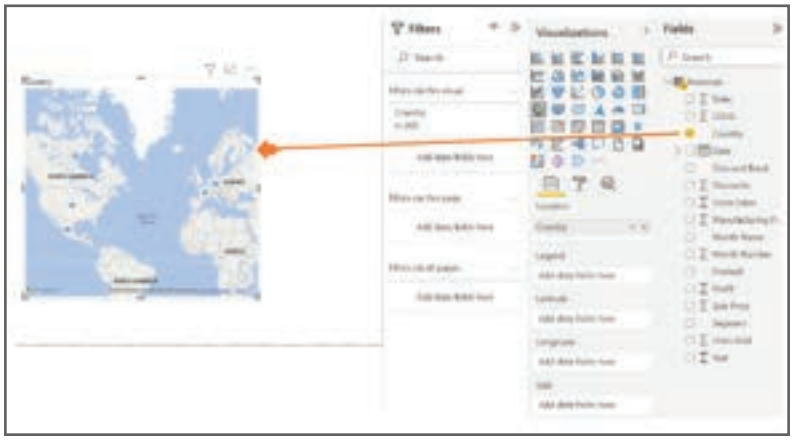


Figure 2.1.5.:Drag a field onto the Report view canvas to create a visual.

For example, Power BI Desktop automatically created a map-based visualization because it recognized that the Country field contained geolocation data.

Publish a report:

After creating a report with a few visuals, you're ready to publish to the Power BI service. On the Home ribbon on the Power BI Desktop, select Publish.

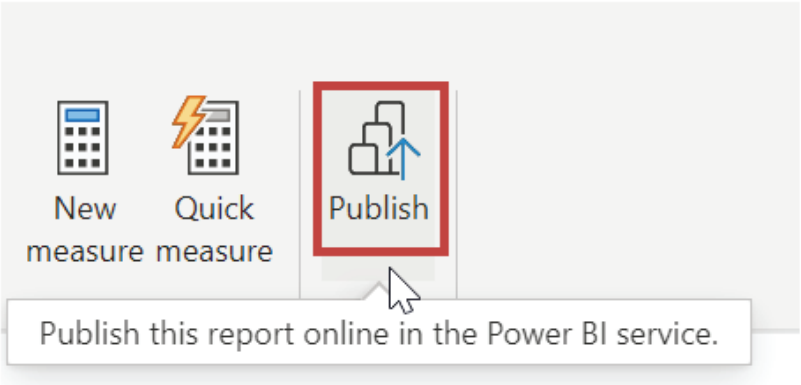


Figure 2.1.6.: Publish button to publish the report online.

You'll be prompted to sign in to Power BI. When you've signed in and the publishing process is complete, the following dialog box will appear. You can select the link below Success!, which will take you to the Power BI service, where you can see the report that you published.

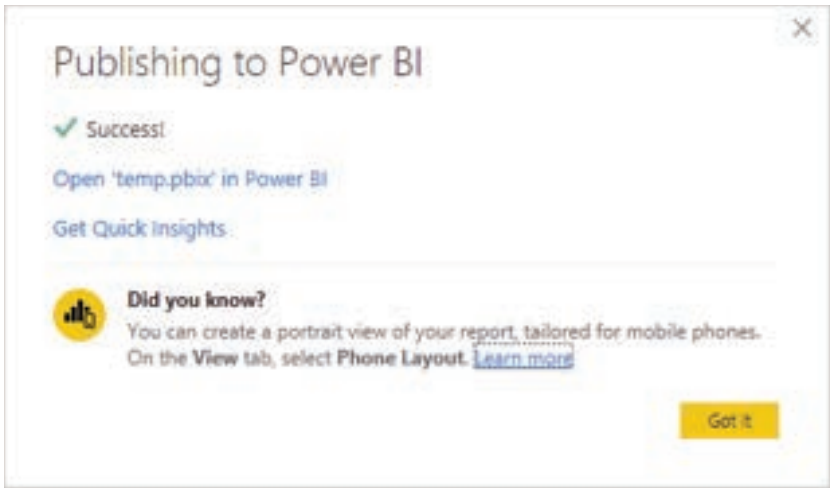


Figure 2.1.7.: Publishing to Power BI success message.

Pin a visual to a dashboard:

When you view a published report in the Power BI service, you can choose the Pin icon to pin that visual to a dashboard.

Figure 2.1.8.: The pin visual button at the top of each visual.

You can choose whether to pin the visual to an existing dashboard or to create a new dashboard.

2.1.3. CONNECT TO DATA SOURCES

Power BI Desktop connects to many types of data sources, including local databases, worksheets, and data on cloud services. Sometimes when you gather data, it's not quite as structured, or clean, as you want it to be. To structure data, you can transform it, meaning that you can split and rename columns, change data types, and create relationships between columns.

In this unit, you will:

- Connect to data.
- Import data into Power BI Desktop.

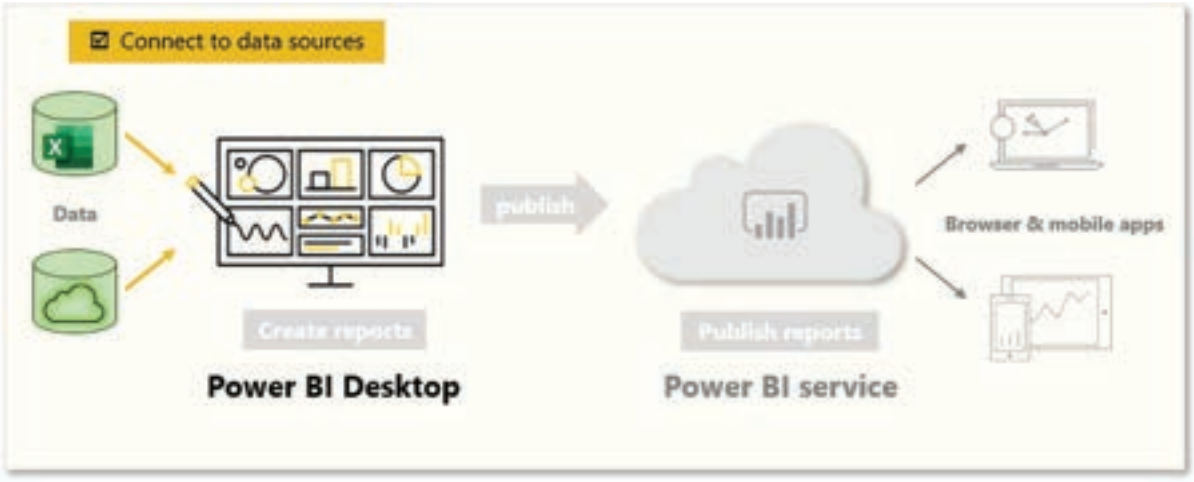


Figure 2.1.9.: Connect to data sources.

You can connect Power BI Desktop to many types of data sources, including on-premises databases, Microsoft Excel workbooks, and cloud services. Currently, there are about 60 Power BI-specific connectors to cloud services such as GitHub and Marketo. You can also connect to generic sources through XML, CSV, text, and ODBC. Power BI will even extract tabular data directly from a website URL.

Connect to data:

When you start Power BI Desktop, you can choose Get Data from the ribbon on the Home tab.

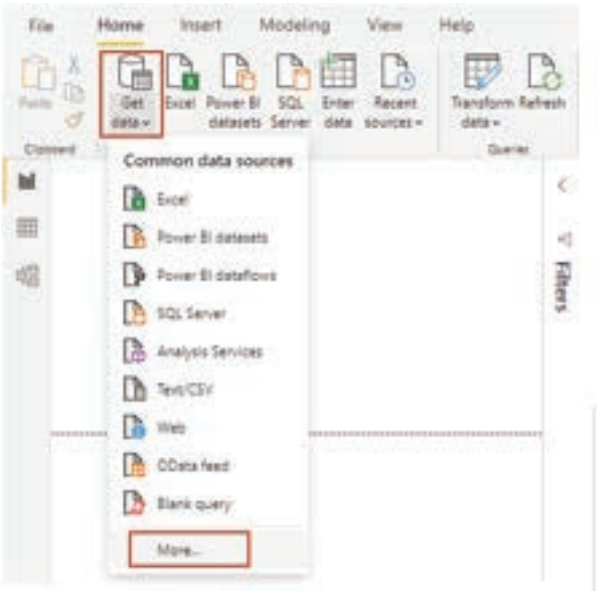


Figure 2.1.10.: Get Data button on the Home tab.

In Power BI Desktop, several types of data sources are available. Select a source to establish a connection. Depending on your selection, you'll be asked to find the source on your computer or network. You might be prompted to sign in to a service to authenticate your request.

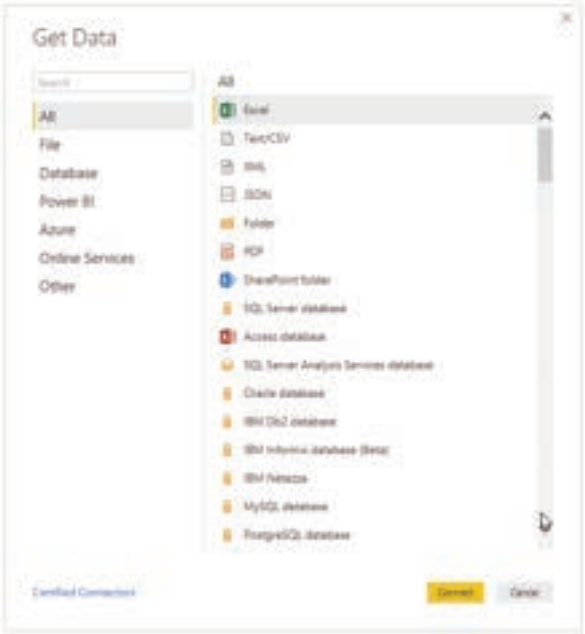


Figure 2.1.11.: Power BI-specific data connectors.

Choose data to import:

After connecting, the first window that you'll see is the Navigator. The Navigator window displays the tables or entities of your data source, and selecting a table or entity gives you a preview of its contents. You can then import your selected tables or entities immediately by selecting Load, or you can select Transform Data to transform and clean your data before importing.

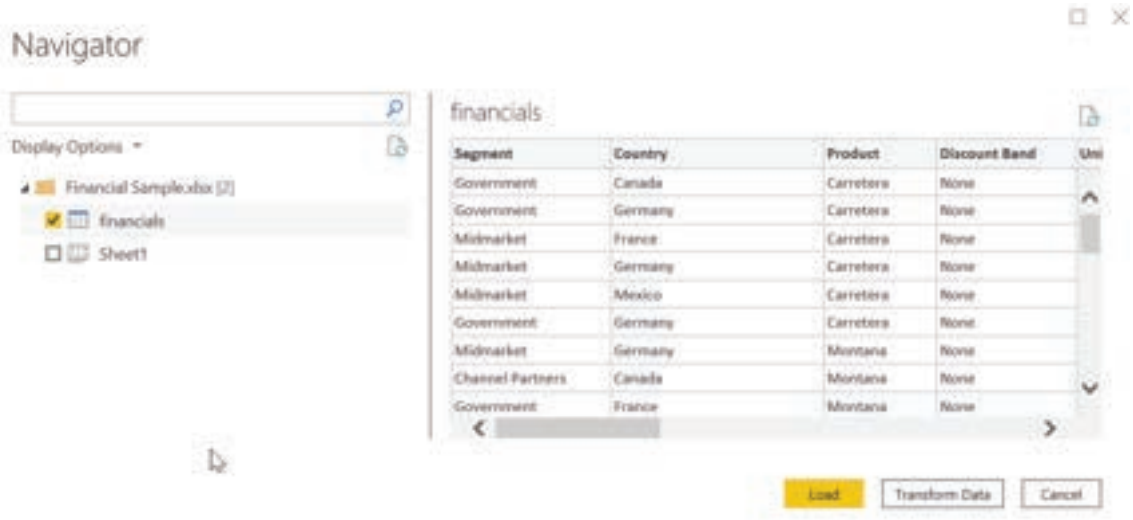


Figure 2.1.12.: Screenshot of the Navigator window.

After you've selected the tables that you'd like to bring into Power BI Desktop, select the Load button. You might want to make changes to those tables before you load them. For example, if you only want a subset of customers or a specific country or region, select the Transform Data button and filter data before loading.

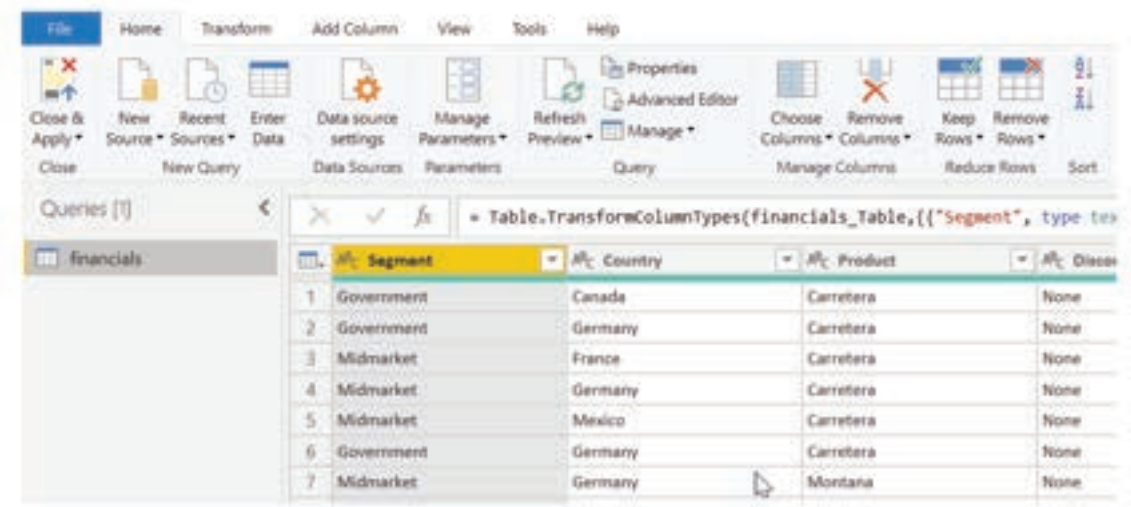


Figure 2.1.13.Screenshot of table data in Edit mode.

No matter what type of data you need, you're likely to find a way to import it into Power BI Desktop.

2.1.4. GET DATA FROM EXCEL

Likely, you've used Microsoft Excel to create or view reports or to build pie charts or other visuals. Getting your Excel data into Power BI is a straightforward process.

In this unit, you will bring Excel workbooks into Power BI.

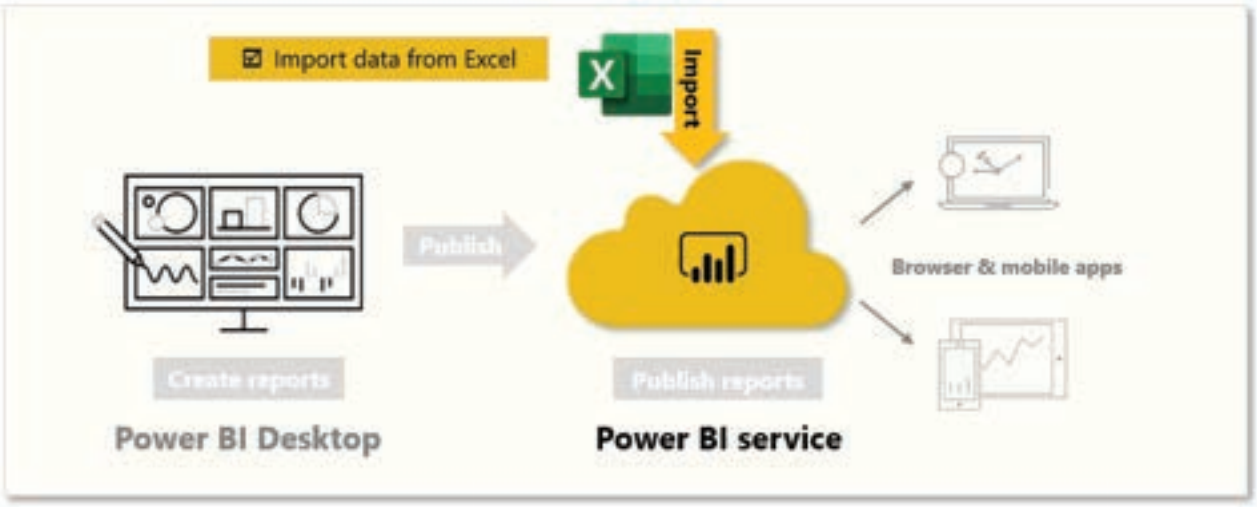


Figure 2.1.14.: Import data from Excel.

This unit explains how you can import an Excel workbook file that contains a simple table from a local drive into Power BI. You'll then learn how to begin exploring that table's data in Power BI by creating a report.

Note:

Up until now, we've been importing data through Power BI Desktop. This unit page is done from the Power BI service.

Make sure that each column has a good name in Excel; it will make it easier for you to find the data that you want when creating your reports in Power BI.

Import from a local drive:

Wherever you keep your files, Power BI makes importing them simple. In Power BI, you can go Get Data > Files > Local File to select the Excel file that you want.

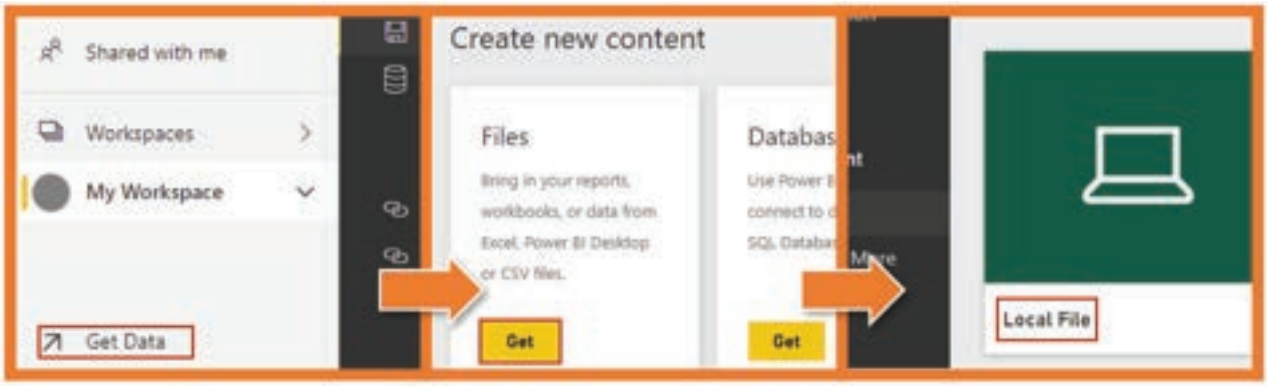


Figure 2.1.15.: Get Data, Get, and the Local file buttons.

After you click Local file, you have two options. You can import Excel data into Power BI or you can upload your Excel file to Power BI.

Import will connect to the data in your workbook so you can create Power BI reports and dashboards. Upload will bring your Excel file into Power BI so you can view and interact with it as you would in Excel Online.

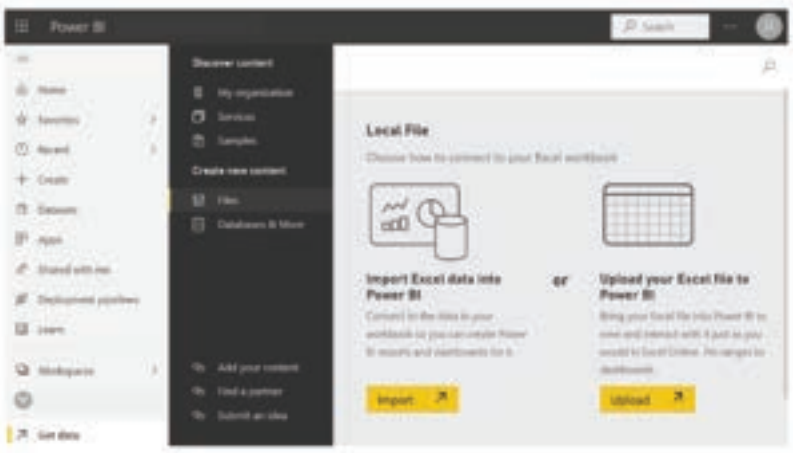


Figure 2.1.16.: Import or upload window.

After the file has been imported into Power BI, you can begin creating reports.

Your files don't have to be on a local drive. If you save your files on OneDrive or SharePoint Team Site, that's even better.

Create reports:

After your workbook's data has been imported, a dataset is created in Power BI and it will appear under Datasets.

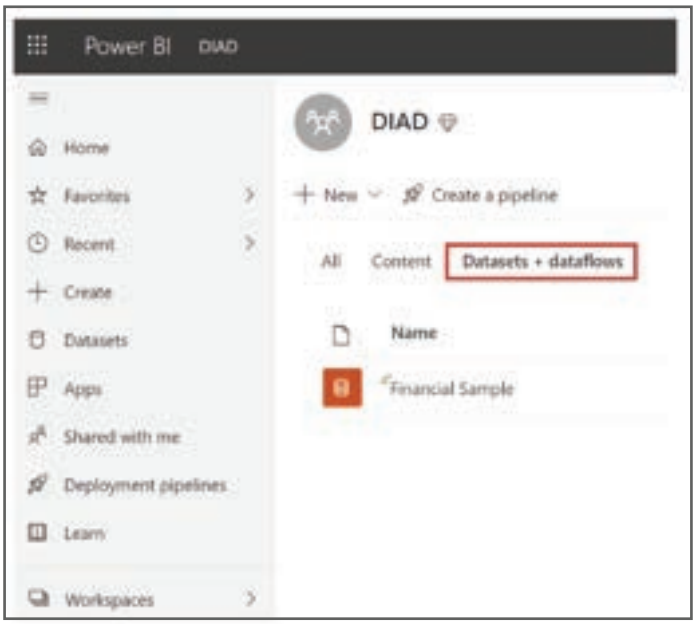


Figure 2.1.17.: My Workspace, Datasets window.

Now, you can begin exploring your data by creating reports and dashboards. Select the (...) icon next to the dataset and then select Create Report. A new blank report canvas appears. On the right-hand side, under Fields, are your tables and columns. Select the fields for which you want to create a new visualization on the canvas.

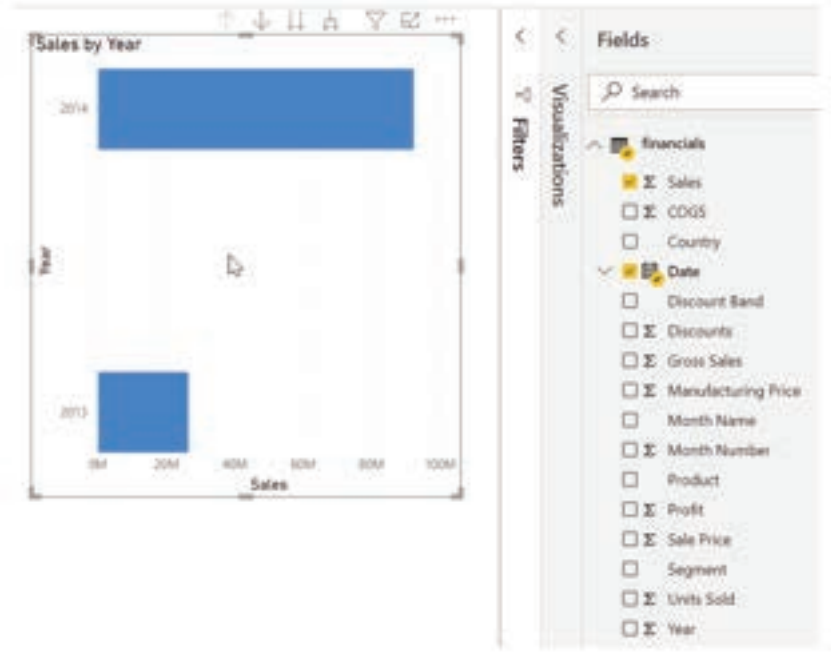


Figure 2.1.18.: Fields pane and the Report view canvas.

You can change the type of visualization and apply filters and other properties under Visualizations.

If you use any of Excel's advanced BI features like Power Query, Power Pivot, or Power View, you can import that data into Power BI, too.

2.1.5. TRANSFORM DATA TO INCLUDE IN A REPORT

Sometimes, your data might contain extra data or have data in the wrong format. Power BI Desktop includes the Power Query Editor tool, which can help you shape and transform data so that it's ready for your models and visualizations.

In this unit, you will transform data with Power Query Editor.

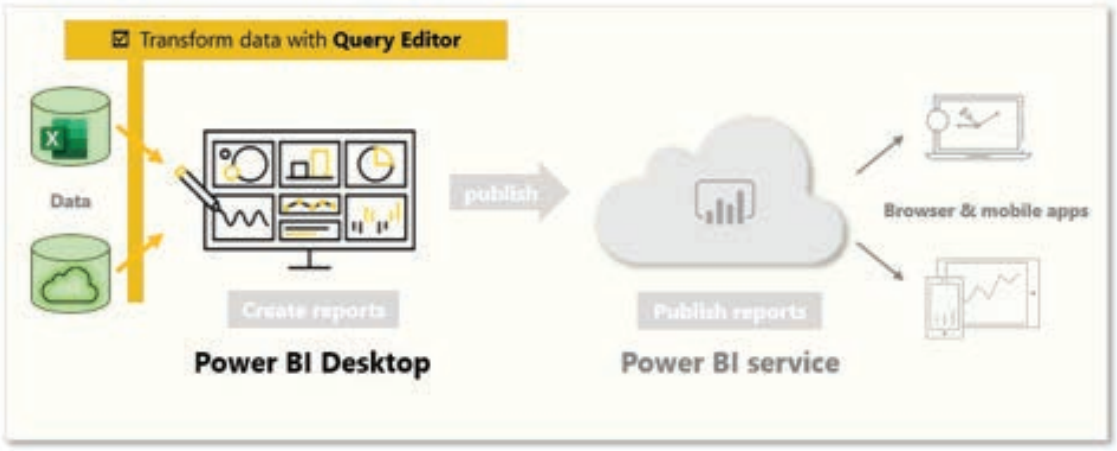


Figure 2.1.19.: Transform data with Query Editor

Launch Power Query Editor:

To begin, select Transform data from the Navigator window to launch Power Query Editor. You can also launch Power Query Editor directly from Power BI Desktop by using the Transform data button on the Home ribbon.

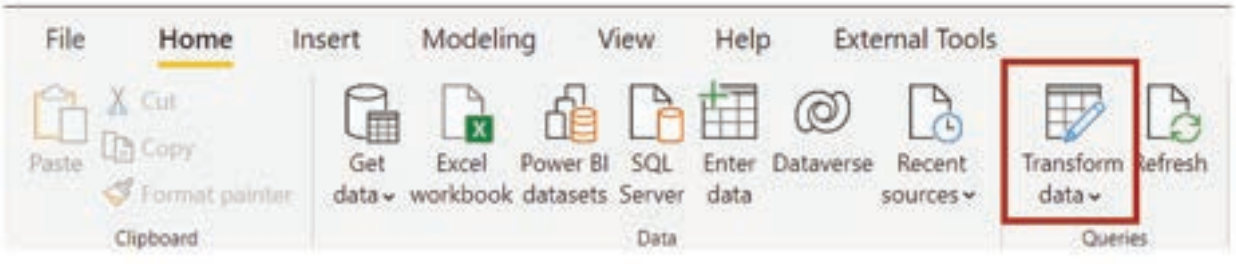


Figure 2.1.20.: Transform data button.

After loading your data into Power Query Editor, you'll see the following screen.

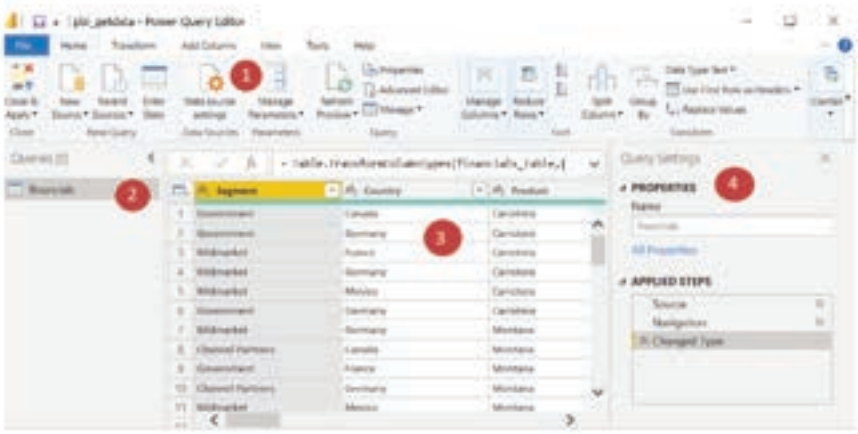


Figure 2.1.21.: The four areas of the Power Query Editor screen.

- In the ribbon, the active buttons enable you to interact with the data in the query.
- On the left pane, queries (one for each table, or entity) are listed and available for selecting, viewing, and shaping.
- On the center pane, data from the selected query is displayed and available for shaping.
- The Query Settings window lists the query's properties and applied steps.

Transform data:

On the center pane, right-clicking a column displays the available transformations. Examples of the available transformations include removing a column from the table, duplicating the column under a new name, or replacing values. From this menu, you can also split text columns into multiples by common delimiters.

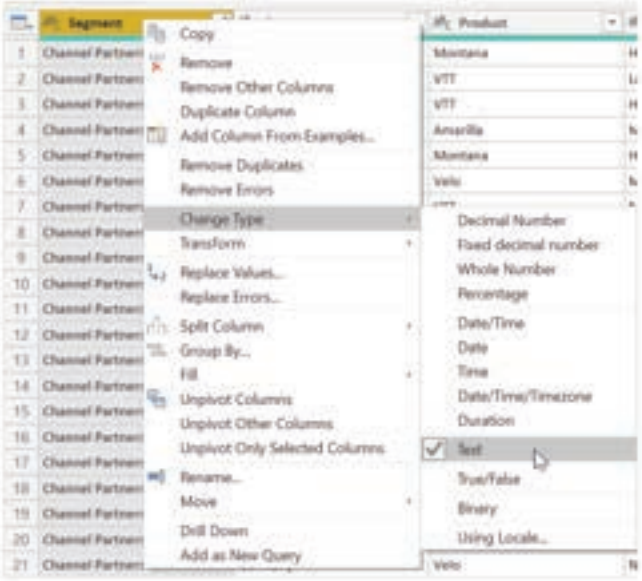


Figure 2.1.22.: Change Type menu.

The Power Query Editor ribbon contains additional tools that can help you change the data type of columns, add scientific notation, or extract elements from dates, such as day of the week.

Tip:

If you make a mistake, you can undo any step from the Applied Steps list.
As you apply transformations, each step appears in the Applied Steps list on the Query Settings pane. You can use this list to undo or review specific changes, or even change the name of a step. To save your transformations, select Close & Apply on the Home tab.

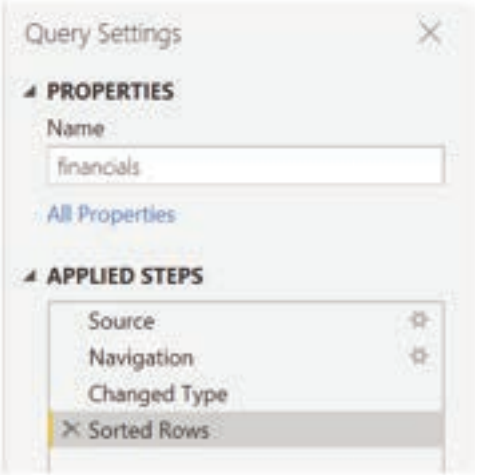


Figure 2.1.23.: Query Settings dialog.

After you select Close & Apply, Power Query Editor applies the query changes and applies them to Power BI Desktop.

2.1.6. COMBINE DATA FROM MULTIPLE SOURCES

With Power BI Desktop, you can use the Power Query Editor tool to combine data from multiple sources into a single report.

In this unit, you will combine data from different sources by using Query Editor.

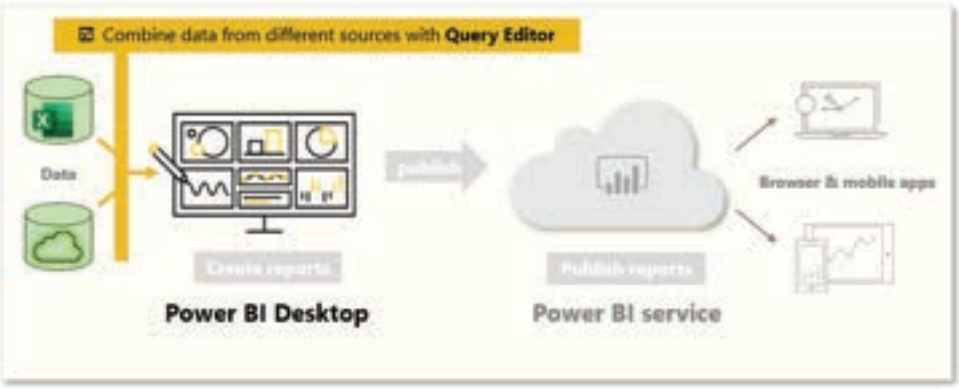


Figure 2.1.24.: Combine data from different sources with Query Editor.

Add more data sources:

To add more sources to an existing report, from the Home ribbon, select Transform data and then select New Source. You can use many potential data sources in Power BI Desktop, including folders. By connecting to a folder, you can import data from multiple Excel or CSV files at once.

Power Query Editor allows you to apply filters to your data. For example, selecting the drop-down arrow next to a column opens a checklist of text filters. Using a filter allows you to remove values from your model before the data is loaded into Power BI.

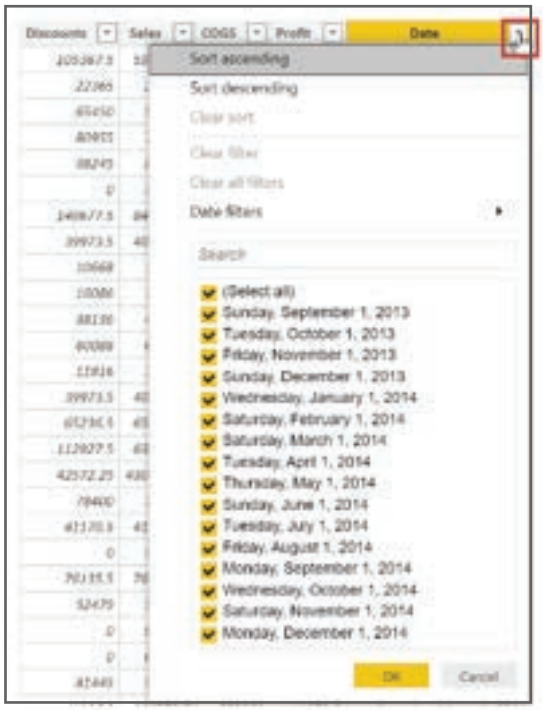


Figure 2.1.25.: Remove Empty option.

Important:

Filtering in the Power Query Editor changes which data is loaded into Power BI. Later, when you apply filters in the Data View or Report View, those filters only apply to what you see in visuals but do not change the underlying dataset.

Merge and append queries:

You can also merge and append queries. In other words, Power BI pulls data that you select from multiple tables or various files into a single table. Use the Append Queries tool to add the data from a new table to an existing query. Power BI Desktop attempts to match the columns in your queries, which you can then adjust as necessary in Power Query Editor.

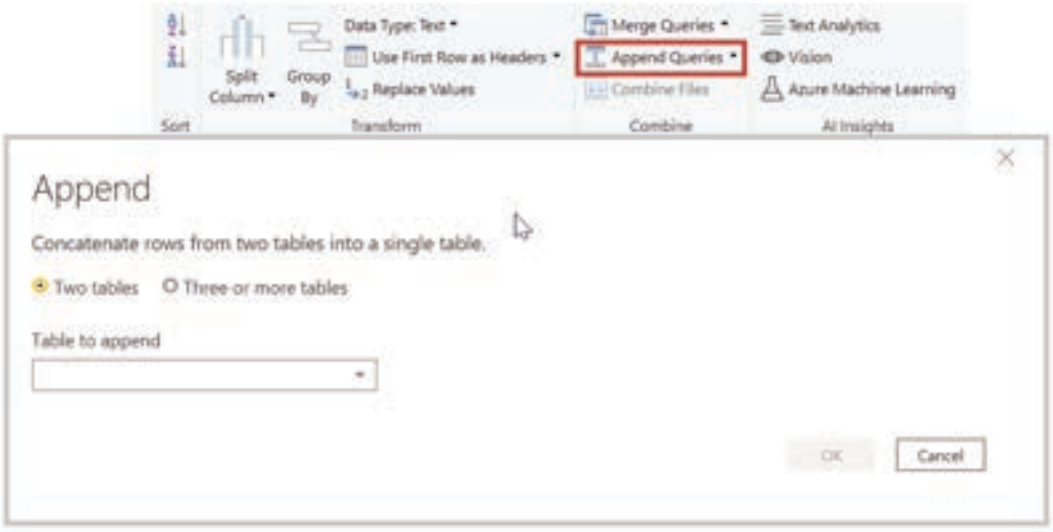


Figure 2.1.26.Append Queries button and dialog.

Write customized queries:

You can use the Add Custom Column tool to write new customized query expressions by using the powerful M language.

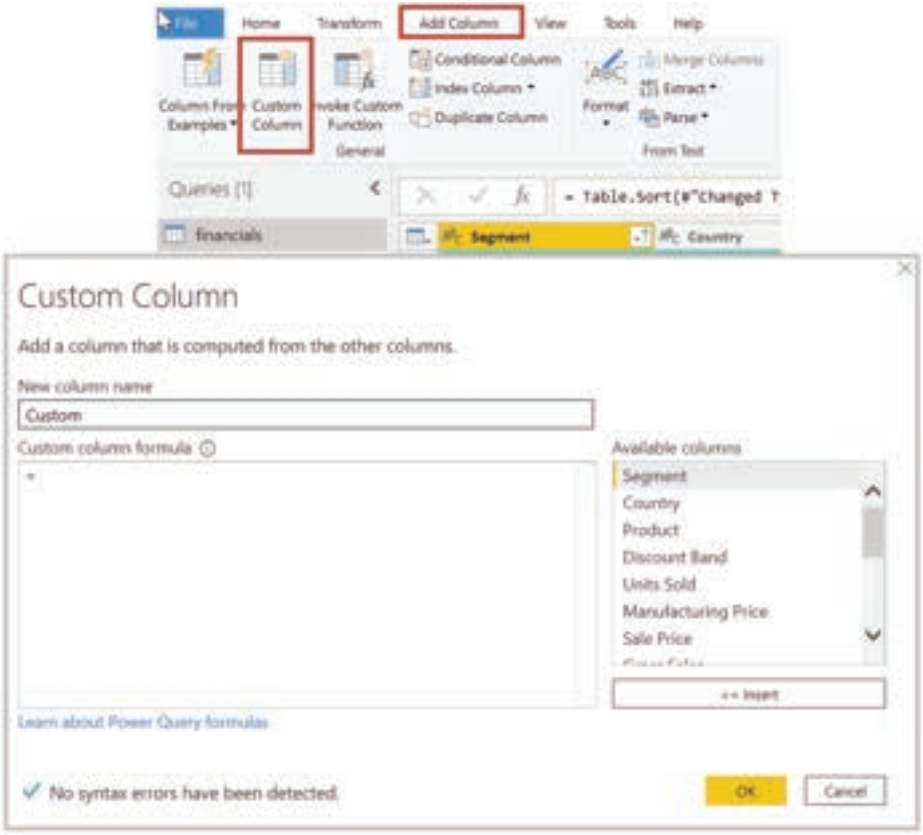


Figure 2.1.27.Custom Column button and dialog.

2.1.7. CLEAN DATA TO INCLUDE IN A REPORT

While Power BI can import your data from almost any source, its visualization and modeling tools work best with columnar data. Sometimes, your data won't be formatted in simple columns, which is often the case with Excel spreadsheets.

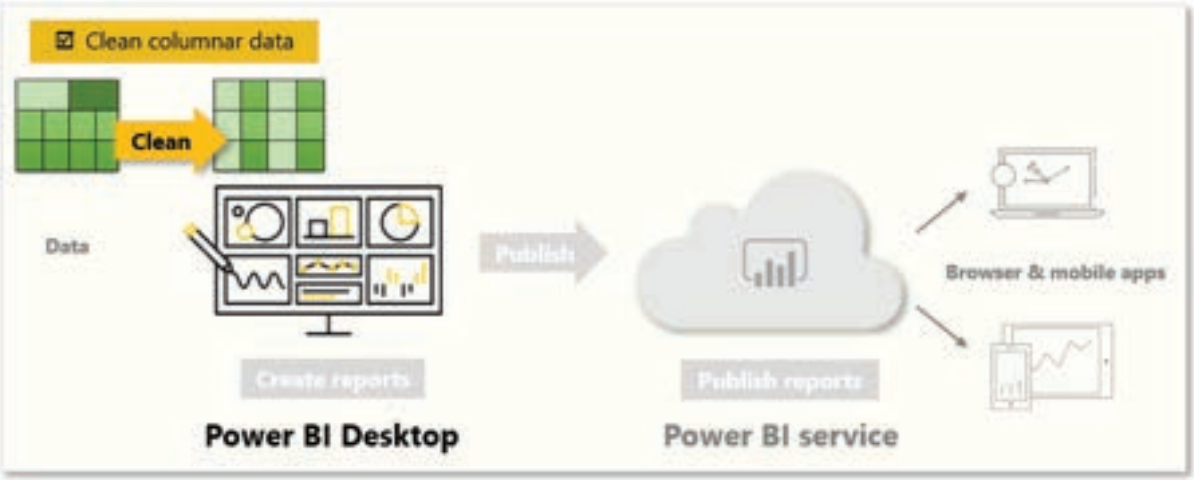
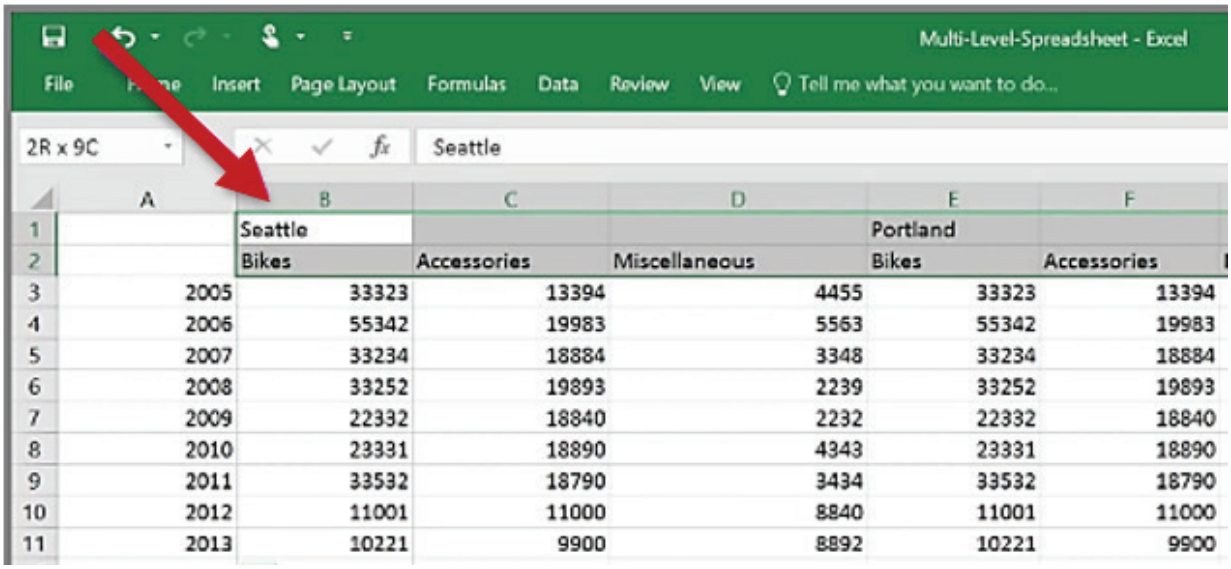


Figure 2.1.28.: Clean columnar data.

A table layout that looks good to the human eye might not be optimal for automated queries. For example, the following spreadsheet has headers that span multiple columns.



	A	B	C	D	E	F
1		Seattle			Portland	
2		Bikes	Accessories	Miscellaneous	Bikes	Accessories
3	2005	33323	13394	4455	33323	13394
4	2006	55342	19983	5563	55342	19983
5	2007	33234	18884	3348	33234	18884
6	2008	33252	19893	2239	33252	19893
7	2009	22332	18840	2232	22332	18840
8	2010	23331	18890	4343	23331	18890
9	2011	33532	18790	3434	33532	18790
10	2012	11001	11000	8840	11001	11000
11	2013	10221	9900	8892	10221	9900

Figure 2.1.29.: Excel spreadsheet with headers that span multiple columns.

Clean data:

Fortunately, Power Query Editor has tools to help you quickly transform multi-column tables into datasets that you can use.

Transpose data:

By using Transpose in Power Query Editor, you can swap rows into columns to better format the data.

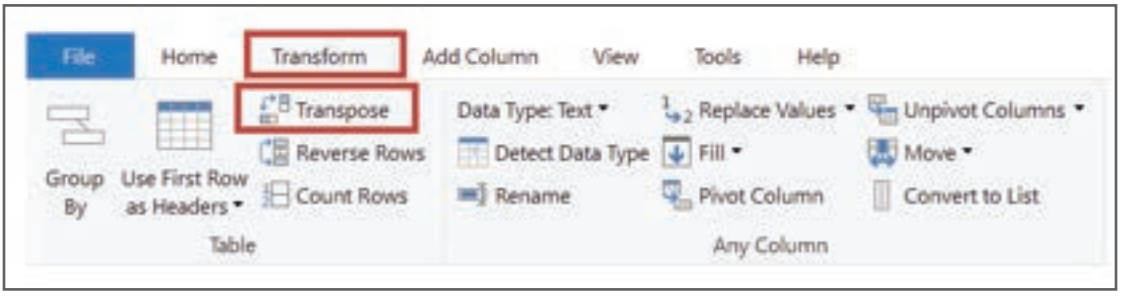


Figure 2.1.30.: Transpose button

Format data:

You might need to format data so that Power BI can properly categorize and identify that data. With some transformations, you'll cleanse data into a dataset that you can use in Power BI. Examples of powerful transformations include promoting rows into headers, using Fill to replace null values, and Unpivot Columns.

With Power BI, you can experiment with transformations and determine which will transform your data into the most usable columnar format. Remember, the Applied Steps section of Power Query Editor records all your actions. If a transformation doesn't work the way that you intended, select the X next to the step, and then undo it.

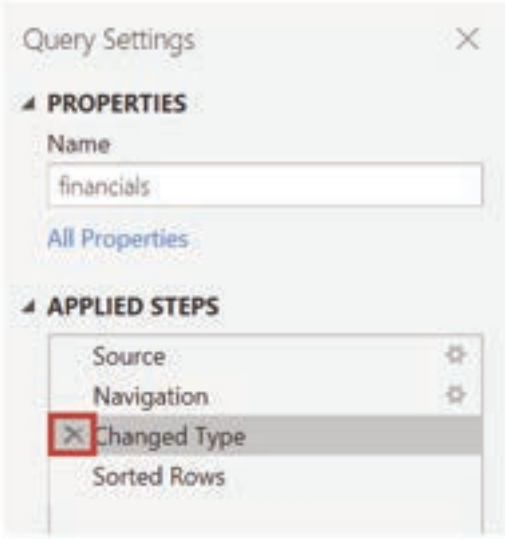


Figure 2.1.31.: How to remove steps from the Applied Steps section.

After you've cleaned your data into a usable format, you can begin to create powerful visuals in Power BI.

SUMMARY

- Power BI Desktop Overview: Users learn about Power BI Desktop's capabilities and its integration with Power BI Service for report sharing.
- Exploring Power BI Desktop: This section provides a hands-on exploration of the user interface, creating visuals, publishing reports, and pinning visuals to dashboards.
- Connecting to Data Sources: Power BI Desktop connects to various data sources, including databases, Excel, and cloud services.
- Getting Data from Excel: Users are guided on importing Excel data into Power BI for report creation.
- Transforming Data: Power Query Editor is introduced for data transformation.
- Combining Data: Users learn to merge data from multiple sources using Power Query Editor.
- Cleaning Data: The importance of data cleanliness for effective visualization is emphasized, with tips on data formatting.

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 What is Power BI Desktop primarily used for?
   a) Creating and editing documents
   b) Gathering, transforming, and visualizing data
   c) Developing mobile applications
   d) Writing code

2 Which component of Power BI allows you to create reports and dashboards?
   a) Power Query Editor
   b) Power BI Service
   c) Power BI Desktop
   d) Power Pivot

3 Where can you download Power BI Desktop from?
   a) Microsoft Store on the Windows tab
   b) App Store on Mac
   c) Google Play Store
   d) It is pre-installed on Windows PCs

4 How can you sign in to Power BI service?
   a) By creating a new account within Power BI Desktop
   b) By signing in with your email address on app.powerbi.com
   c) By using a QR code
   d) By visiting a physical Microsoft store

5 What is the first step in creating a visual in Power BI Desktop?
   a) Selecting the visual type
   b) Dragging a field onto the Report view canvas
   c) Publishing the report
   d) Transforming the data

6 What is the purpose of the Visualizations pane in Power BI Desktop?
   a) To create visuals
   b) To connect to data sources
   c) To change visualizations, customize colors, and apply filters
   d) To launch Power Query Editor

7 How can you publish a report to the Power BI service from Power BI Desktop?
   a) Clicking the Publish button on the Home ribbon
   b) Clicking the Transform Data button
   c) Clicking the Create Report button
   d) Clicking the Export as PDF button

8 What does the "Pin a visual to a dashboard" feature in Power BI service allow you to do?
   a) Share the visual on social media
   b) Print the visual
   c) Add the visual to a dashboard
   d) Export the visual as an image

9 How can you connect to different types of data sources in Power BI Desktop?
   a) By typing the data source URL
   b) By selecting "Transform Data" from the File menu
   c) By clicking "Get Data" on the Home tab
   d) By using the Visualizations pane

10 Which window in Power BI Desktop displays the tables or entities of your data source and allows you to preview their contents?
   a) Query Settings
   b) Visualizations pane
   c) Navigator
   d) Report view canvas

11 What does the "Transform Data" option in Power BI Desktop allow you to do before importing data?
   a) Create reports
   b) Filter data
   c) Rename columns
   d) Apply filters

12 How can you import Excel workbooks into Power BI?
   a) By uploading them directly to the Power BI service
   b) By copying and pasting the data
   c) By selecting "Transform Data" in Excel
   d) By using the "Get Data" option in Power BI Desktop

13 Which tool in Power Query Editor allows you to write custom query expressions?
   a) Power Pivot
   b) Power Query
   c) M Language Editor
   d) Add Custom Column

14 What is the purpose of transposing data in Power Query Editor?
   a) To import data from multiple sources
   b) To change the data type of columns
   c) To swap rows into columns
   d) To create reports

15 How can you combine data from multiple sources into a single report in Power BI Desktop?
   a) By using Power Pivot
   b) By importing data separately into each visual
   c) By using the Append Queries tool in Query Editor
   d) By creating a new Power BI Desktop file for each source

Answers

1 *B) Gathering, transforming, and visualizing data*
2 *C) Power BI Desktop*
3 *A) Microsoft Store on the Windows tab*
4 *B) By signing in with your email address on app.powerbi.com*
5 *B) Dragging a field onto the Report view canvas*
6 *C) To change visualizations, customize colors, and apply filters*
7 *A) Clicking the Publish button on the Home ribbon*
8 *C) Add the visual to a dashboard*
9 *C) By clicking "Get Data" on the Home tab*
10 *C) Navigator*

11 *B) Filter data*
12 *D) By using the "Get Data" option in Power BI Desktop*
13 *D) Add Custom Column*
14 *C) To swap rows into columns*
15 *C) By using the Append Queries tool in Query Editor*

SELF-EXAMINATION QUESTIONS FOR PRACTICE:

1 What is the primary purpose of Power BI Desktop, and how does it integrate with the Power BI Service?

2 Can you name at least three different types of data sources that Power BI Desktop can connect to?

3 In Power BI Desktop, how do you create a new visual on the report canvas? Provide a step-by-step process.

4 What is the role of Power Query Editor in data analysis within Power BI Desktop, and why is it important?

5 Explain the concept of "cleaning data" in the context of Power BI, and provide an example of a data cleaning operation you might perform.

6 How can you combine data from multiple sources in Power BI Desktop, and why might you need to do this in a real-world data analysis scenario?

7 What are the key advantages of using Power BI Desktop over traditional spreadsheet software like Microsoft Excel for data analysis and visualization?

8 Describe the steps involved in publishing a report from Power BI Desktop to the Power BI Service.

9 How can you transform data using the Power Query Editor, and why is data transformation important for effective analysis?

10 When should you consider using the "Transpose" function in Power Query Editor, and what benefits does it offer in data preparation?

CHAPTER 3
QUERYING AND SHAPING THE DATA:

The first step to insights begins in Power BI Desktop where you connect to and transform your data in preparation for data modeling. Start here to learn how to ingest and transform data in Power BI.

3.1. GET DATA IN POWER BI:

LEARNING OBJECTIVES

- 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

You'll learn how to retrieve data from a variety of data sources, including Microsoft Excel, relational databases, and NoSQL data stores. You'll also learn how to improve performance while retrieving data.

3.1.1. INTRODUCTION

Like most of us, you work for a company where you're 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 that contains what items each customer bought 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 Cosmos DB, 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're asked to combine data from.

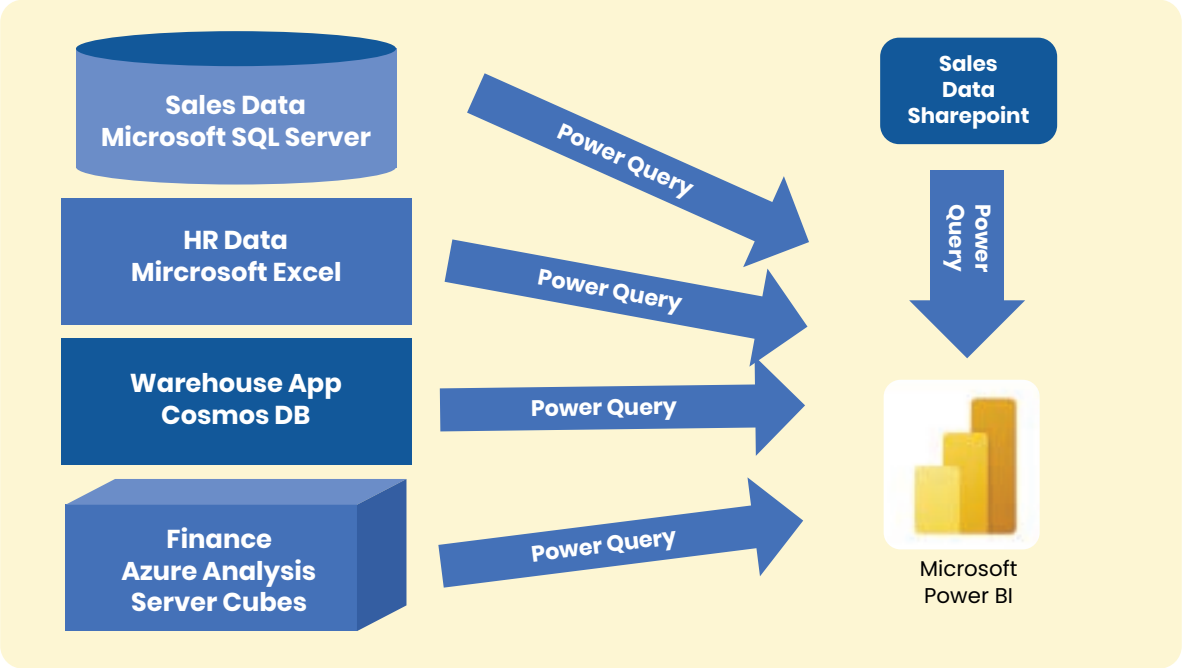


Figure 3.1.1.: Power Query delivering data from various locations to Power B I.

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 gaining understanding of the systems, you can use Power Query to help you clean the data, such as renaming columns, replacing values, removing errors, and combining query results. Power Query is also available in Excel. After the data has been cleaned and organized, you're ready to build reports in Power BI. Finally, you'll publish your combined dataset and reports to Power BI service. From there, other people can use your dataset and build their own reports or they can use the reports you've already built. Additionally, if someone else built a dataset 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.

3.1.2. GET DATA FROM 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 doesn't 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).

Figure 3.1.2.: Data from flat files icons.

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 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 aren't 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.

Figure 3.1.3.: Getting data from files stored locally or from the cloud OneDrive or SharePoint.

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 doesn't 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'll select Excel.

Tip:
The Home tab contains quick access data source options, such as Excel, next to the Get data button.

Figure 3.1.4.: Home Ribbon gets data dropdown menu select excel.

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'll open the Employee Data Excel workbook that is stored on the Desktop (Remember, no files are provided for practice, these are hypothetical steps).

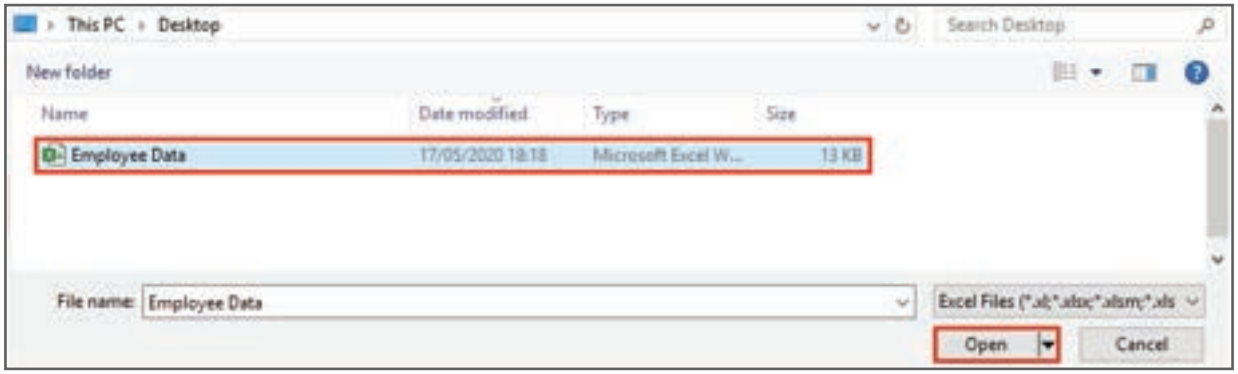


Figure 3.1.5.: Selecting the file called employee data stored on the desktop.

Select the file data to import:

After the file has been 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.

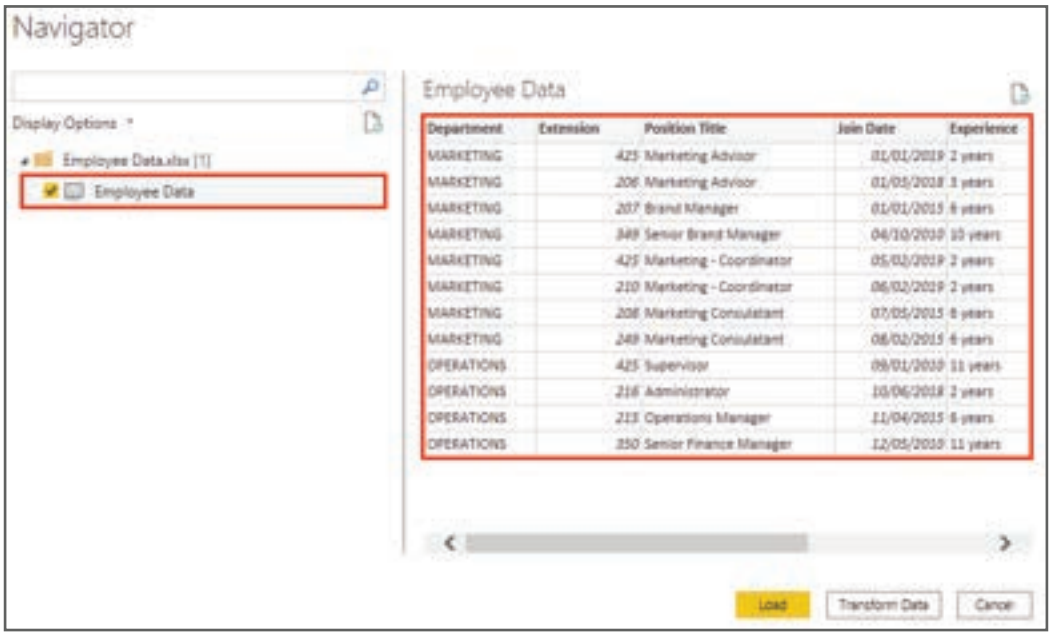


Figure 3.1.6.: Navigator window in Power B I Desktop.

Now you can 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 many ways for you to accomplish this task, so that you can make this type of change when needed.

Data source settings

Query settings

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.

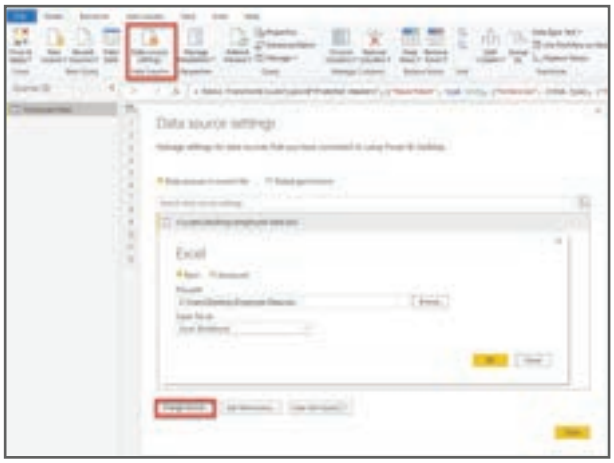


Figure 3.1.7.: Data Source settings window in Power BI Desktop.

3.1.3. GET DATA FROM RELATIONAL DATA SOURCES

If your organization uses a relational database for sales, you can use Power BI Desktop to connect directly to the database instead of using exported 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 has 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.

Figure 3.1.8.: Data flow from SQL database into Power BI.

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.

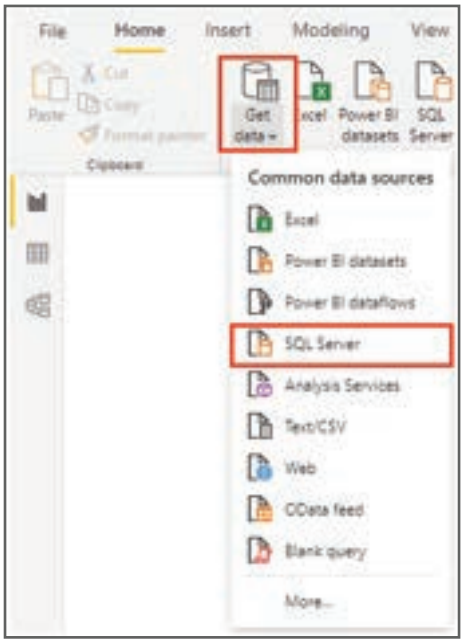


Figure 3.1.9.: Get Data menu expanded to show SQL Server.

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.

Figure 3.1.10.: SQL Server database details

After you've added your server and database names, you'll be prompted to sign in with a username and password. You'll 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.

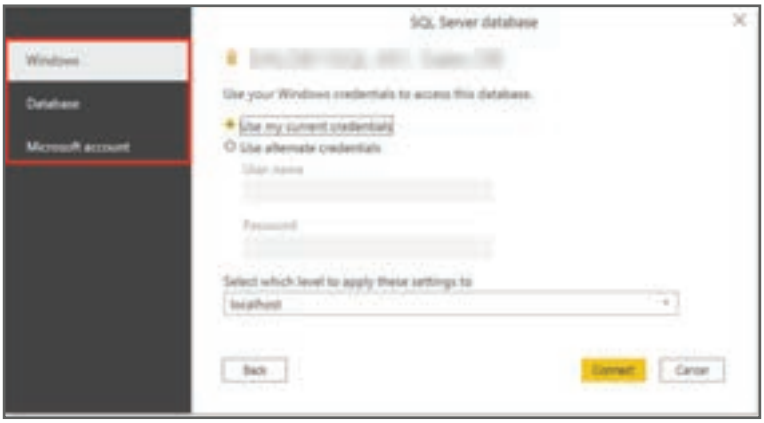


Figure 3.1.11.: Database authorization details.

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.

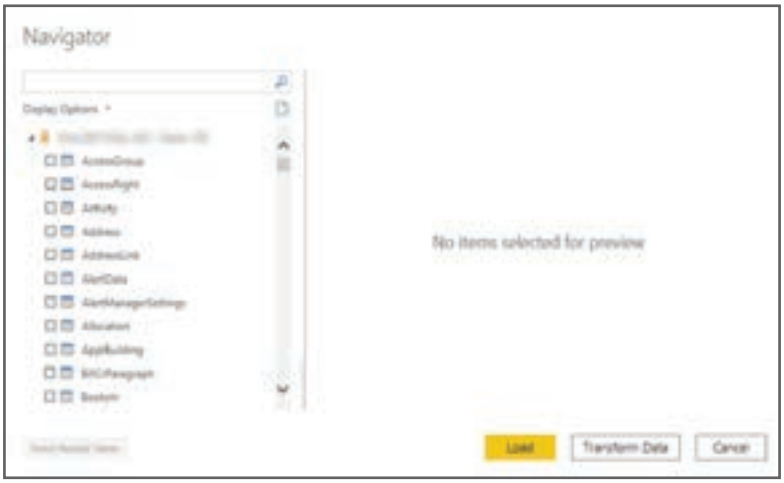


Figure 3.1.12.: Navigator window with available tables.

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'll use the Select SQL statement to load the ID, NAME and SALESAMOUNT columns from the SALES table.

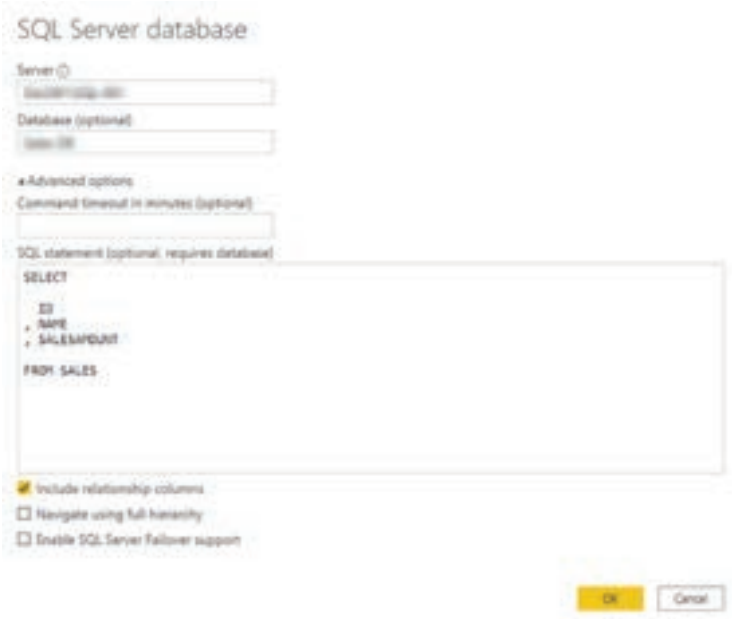


Figure 3.1.13.: SQL Server database dialog with a SQL query.

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.

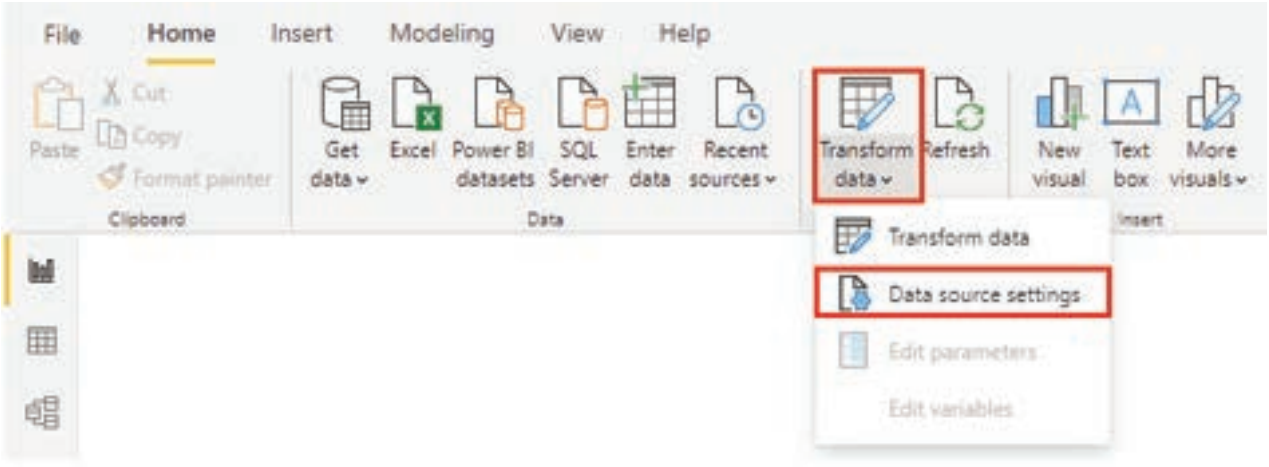


Figure 3.1.14.: Transform data menu expanded with Data source settings highlighted.

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.

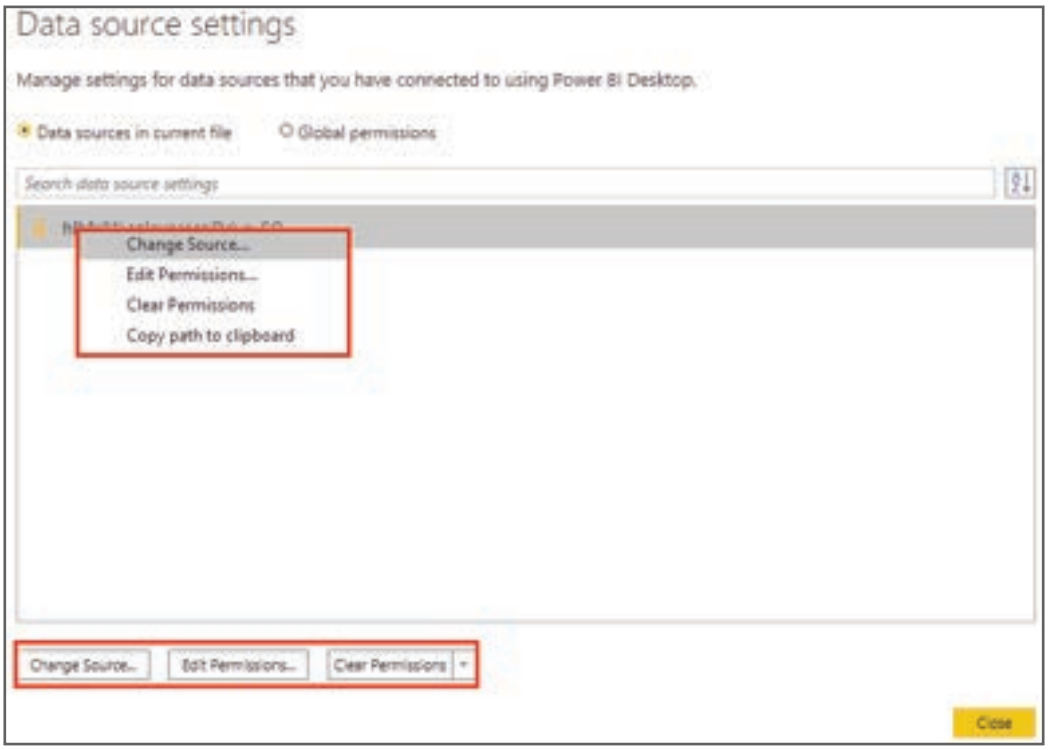


Figure 3.1.15.: Data source settings options.

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.

Figure 3.1.16.: Data source settings button.

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 isn't relevant to the report that you're 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
```

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 January 1st, 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's 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.

3.1.4. CREATE 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 because 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 users more power over the data that is displayed in your reports; they can change the data source and filter the data by themselves.

In the following example, you've created a report for the Sales team at Tailwind Traders that displays the sales data in the SQL Server database. The report gives a holistic view of how the Sales team is performing. While the report is useful, the Sales team members want to be able to filter the report so that they can view their own data only and 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. Then 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 by following these steps:

1 After you have entered your server details, in the SQL Server database window, select Advanced options.

2 Paste the SQL query into the SQL statement box and then select OK.

```
SELECT [SalesOrderID]
      ,[OrderDate]
      ,[OnlineOrderFlag]
      ,[SalesOrderNumber]
      ,[CustomerID]
      ,[SalesPersonID]
      ,[TerritoryID]
      ,[SubTotal]
      ,[TaxAmt]
      ,[Freight]
      ,[TotalDue]
  FROM [TailwindTraders].[Sales].[SalesOrderHeader] s
  WHERE s.[SalesPersonID] = 279
```

Figure 3.1.17.: SQL Query Details

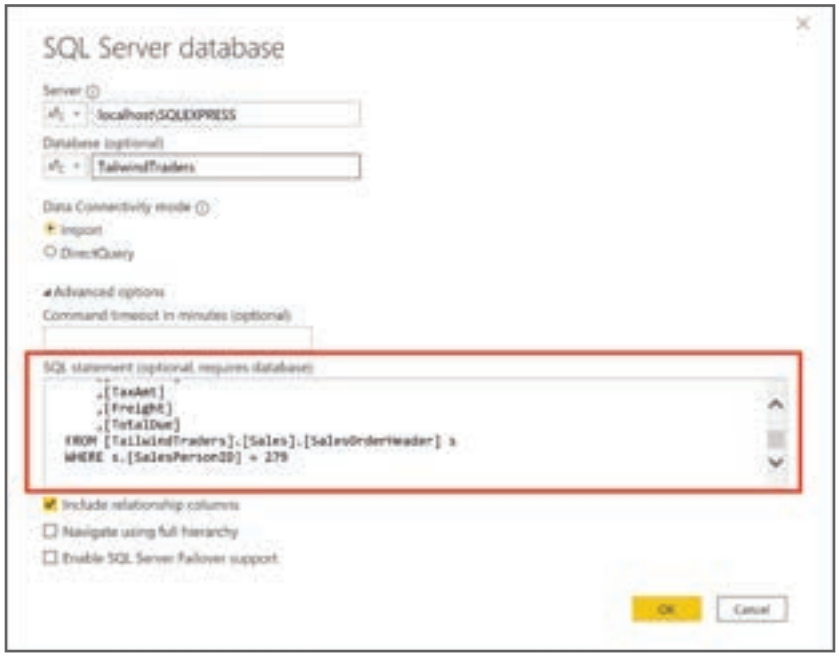


Figure 3.1.18.: Add query to execution statement

3 Select Edit to open the data in Power Query Editor.

Next, you create the parameter by following these steps:

i On the Home tab, select Manage parameters > New parameter.
ii On the Parameters window, change the default parameter name to something more descriptive so that its purpose is clear. In this case, you change the name to SalesPerson.
iii Select Text from the Type list and then select Any value from the Suggested value list.
iv Select OK.

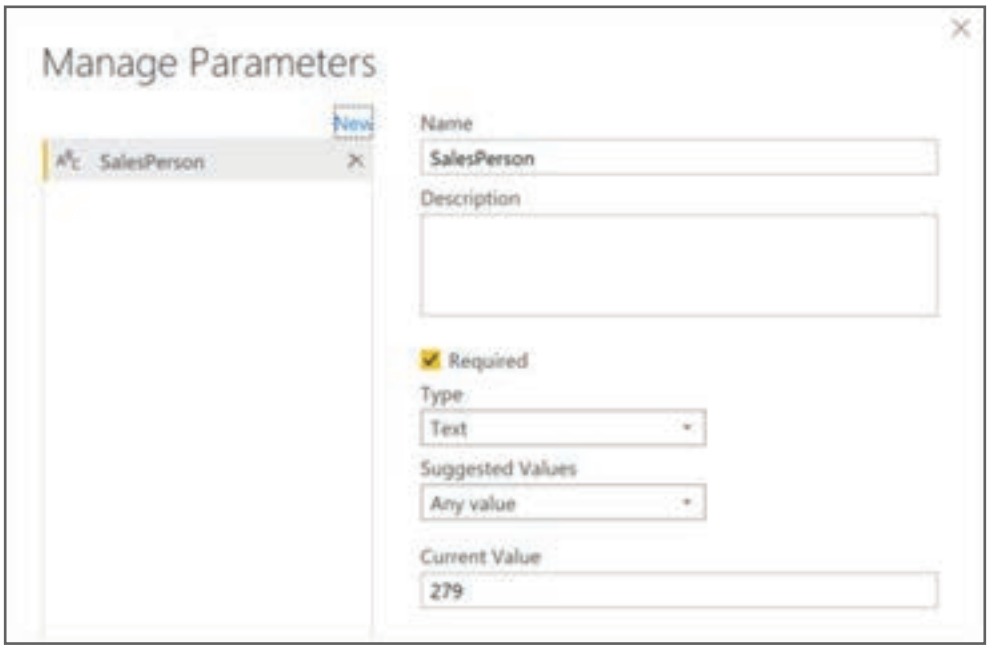


Figure 3.1.19.: Add parameter

A new query is shown for the parameter that you created.

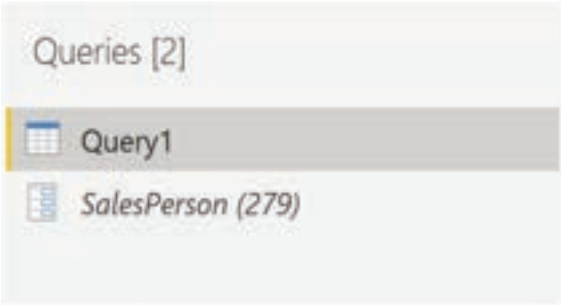


Figure 3.1.20.: New query for parameter

Now, you need to adjust the code in SQL query to assess your new parameter:

i Right-click Query1 and then select Advanced editor.

ii Replace the existing value in the execute statement with an ampersand (&) followed by your parameter name (SalesPerson), as illustrated in the following image.

Figure 3.1.21.: Adjust SQL query statement

iii Make sure that no errors are shown at bottom of the window and then select Done.
Though you don't see a difference on the screen, Power BI ran the query.

iv To confirm that the query was run, you can run a test by selecting the parameter query and entering a new value in the Current Value box.

Figure 3.1.22.: Enter value into parameter

v A warning icon might display next to the query. If so, select that query to view the warning message, which states that permission is required to run this native database query. Select Edit Permission and then select Run.

When the query runs successfully, the parameter 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

Figure 3.1.23.: Parameter with updated values

vi Select Close and Apply to return to the report editor.

Now, you can apply the parameter to the report:

a Select Edit queries > Edit parameters.
b On the Edit Parameters window, enter a new value and then select OK.
c Select Apply changes and then run the native query again.

Now, when you view the data, you see the data for the new value that was passed through the parameter.

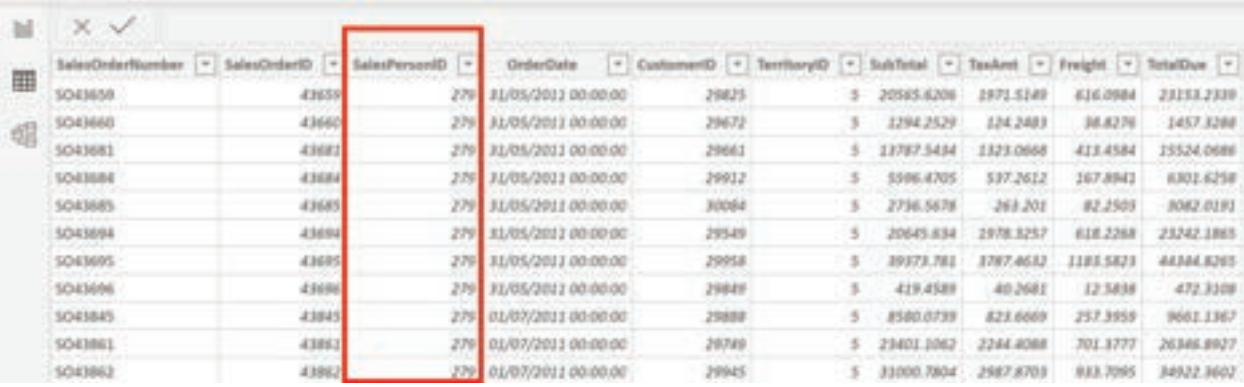


Figure 3.1.24.: Result of applying parameter to report

You can now create a report that displays data for one particular value at a time. More steps are needed to display data for multiple values at the same time.

Create dynamic reports for multiple values:

To accommodate multiple values at a time, you first need to create a Microsoft Excel worksheet that has a table consisting of one column that 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 then follow these steps:

1 On the Navigator window, select Edit to open the data in Power Query Editor, where you see a new query for the data table.

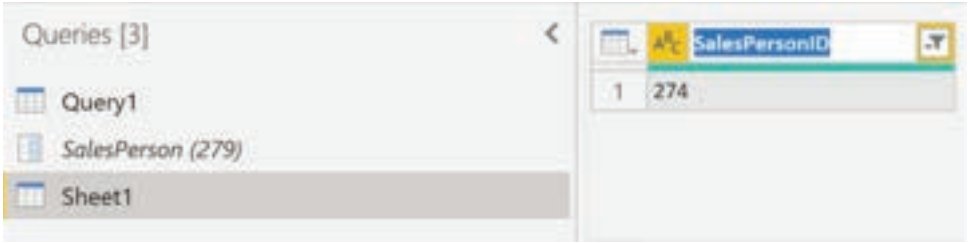


Figure 3.1.25.: Table in Query pane

1 Rename the column in the table to something more descriptive.
2 Change the column data type to Text so that it matches the parameter type and you avoid data conversion problems.
3 In the query Properties section, change the name of the data source to something more descriptive. For this example, enter SalesPersonID.

Next, you need to create a function that passes the new SalesPersonID query into Query1:

i Right-click Query1 and then select Create function.

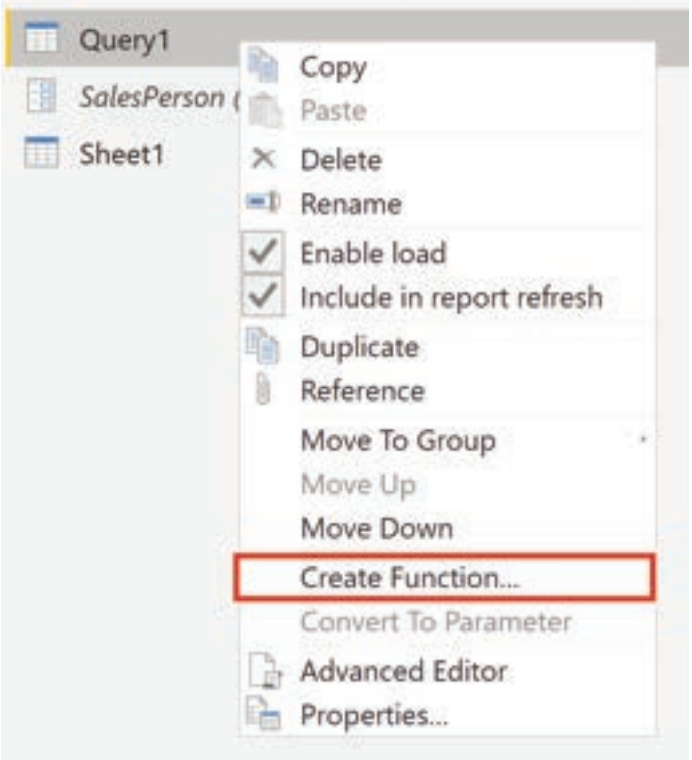


Figure 3.1.26.: Select create function option for query

ii Enter a name for the function and then select OK.

Figure 3.1.27.: Select create function window

Your new function appears in the Queries pane.

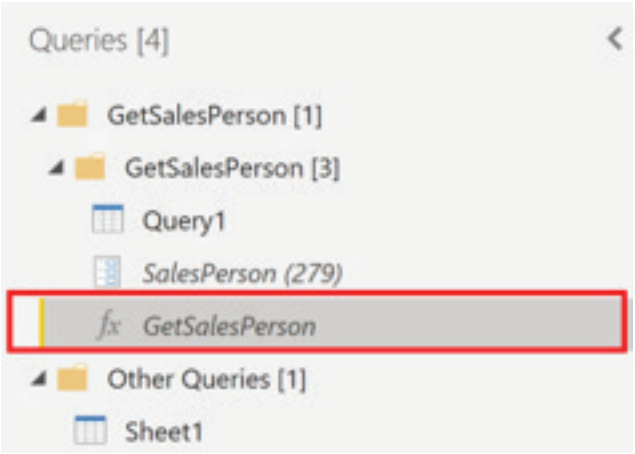


Figure 3.1.28.: Function in Query pane

iii To ensure that Query1 doesn't show up in the field list for the report, which could potentially confuse users, you can disable it from loading in the report by right-clicking Query1 again and then selecting Enable load (selected by default) to disable the feature.

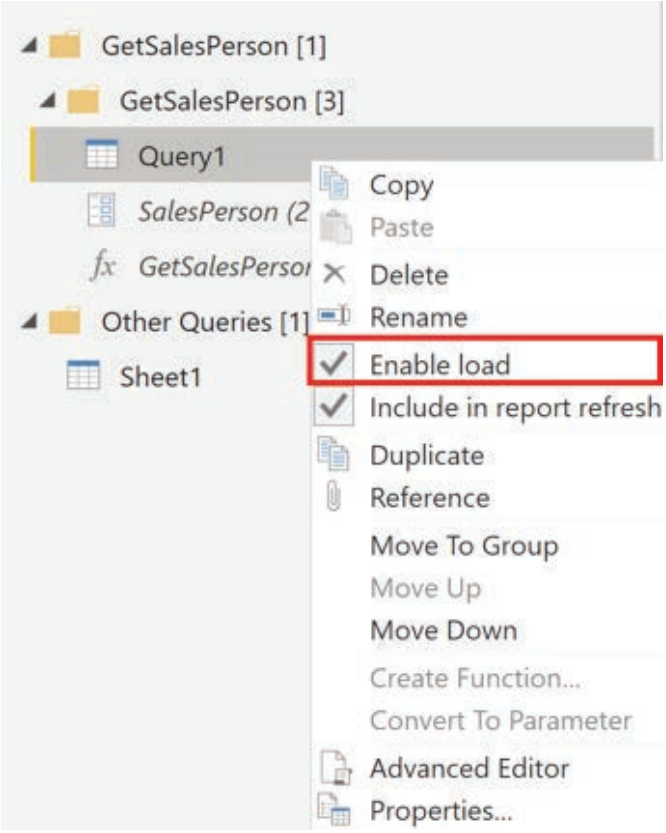


Figure 3.1.29.: Enable load option.

iv Select the SalesPersonID query that you loaded from the Excel worksheet and then, on the Add Column tab, select Invoke custom function to run the custom function that you created.

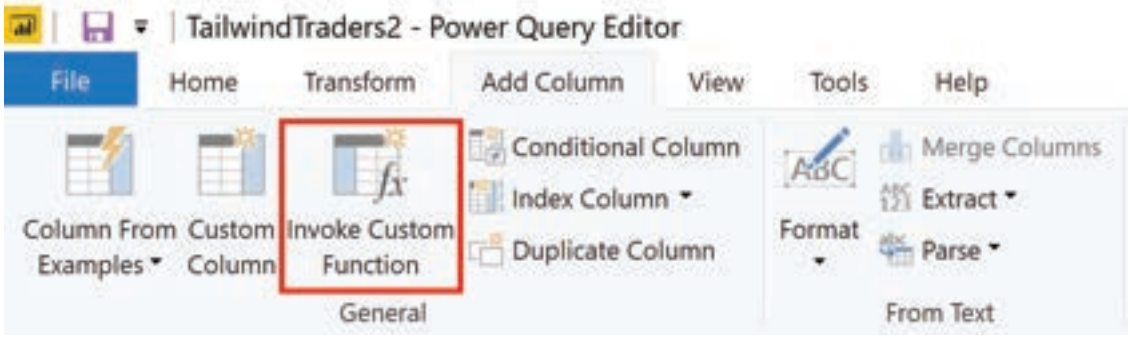


Figure 3.1.30.: Invoke custom function option

v On the Invoke Custom Function window, select your function from the Function query list.

The New column name updates automatically and the table that contains the values that you're going to pass through the parameter is selected by default.

vi Select OK and, if necessary, run the native query.

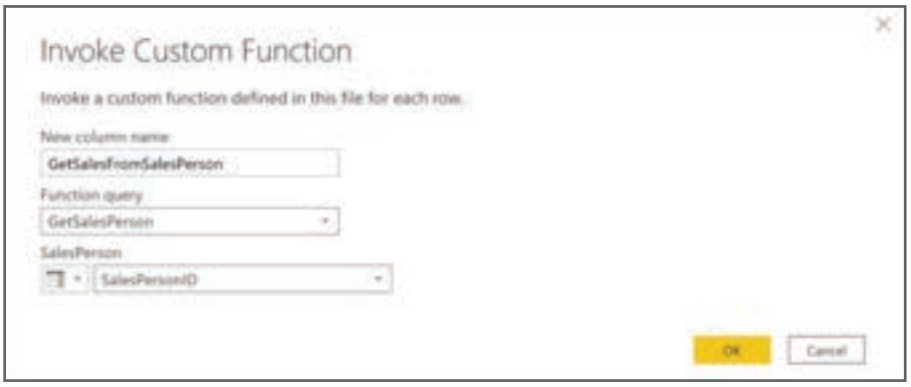


Figure 3.1.31.: Invoke custom function window.

A new column for your GetSalesFromSalesPerson function appears next to the SalesPersonID column.

Figure 3.1.32.: New column for function.

vii Select the two arrows icon in the new column header and then select the check boxes of the columns that you want to load. This section is where you determine the details that are available in the report for each value (sales person ID).

viii Clear the Use original column name as prefix check box at the bottom of the screen because you don't need to see a prefix with the column names in the report.

ix Select OK.

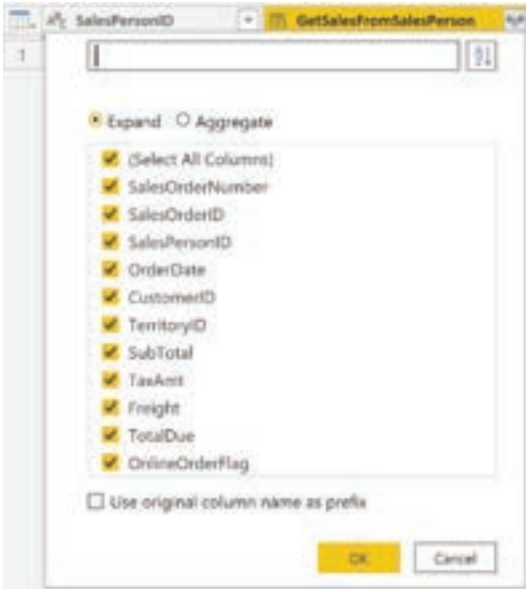


Figure 3.1.33.: Select columns for function

You should be able to view the data for the columns that you selected, for each value (sales person ID).

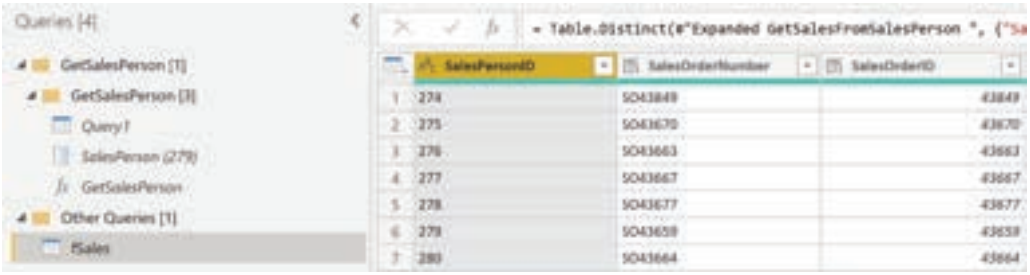


Figure 3.1.34.: View columns for function

If necessary, you can add more values (sales people IDs) to the SalesPersonID column in the Excel worksheet, or you can change the existing values.

x Save your changes and then return to Power Query Editor.

xi On the Home tab, select Refresh Preview, and then run the native query again (if necessary). You should see the sales from the new sales people IDs that you added into the worksheet.

xii Select Close and Apply to return to the report editor, where you see the new column names in the Fields pane.
Now, you can start building your report.

3.1.5. GET DATA FROM A NOSQL DATABASE

Some organizations 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 doesn't use tables to store data.

Scenario:

Software developers at Tailwind Traders created an application to manage shipping and tracking products from their warehouses. The application uses Cosmos DB, a NoSQL database, as the data repository. Data is stored as 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'll use the Get data feature in Power BI Desktop. However, this time you'll select the More... option to locate and connect to the type of database that you use. In this example, you'll select the Azure category, select Azure Cosmos DB, and then select Connect.

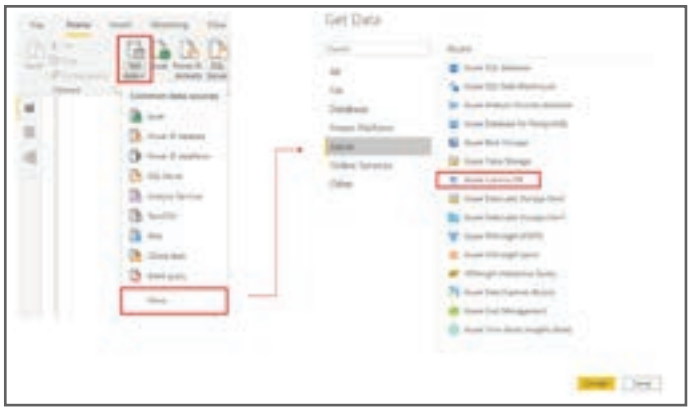


Figure 3.1.35.: Get Data from Azure Cosmos D B option.

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're 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:

If you're working with data stored in JSON format, it's often necessary to extract and normalize the data first. This is because JSON data is often stored in a nested or unstructured format, which makes it difficult to analyze or report on directly.

In this example, the data 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've 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'll 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.

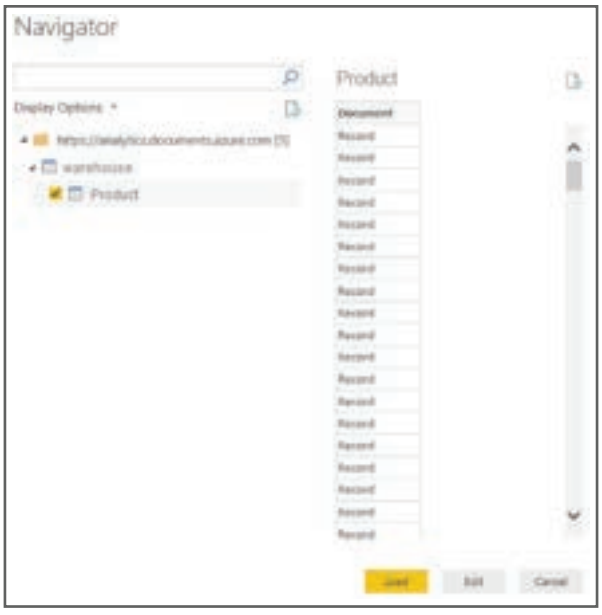


Figure 3.1.36.: Navigator window with list of available records.

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 displays 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.

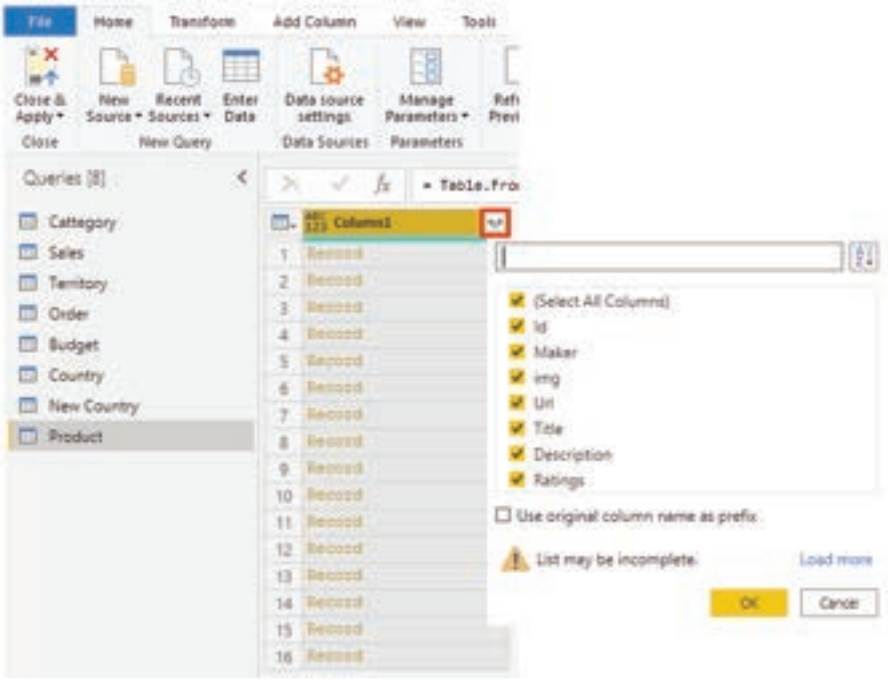


Figure 3.1.37.: Expand records function.

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

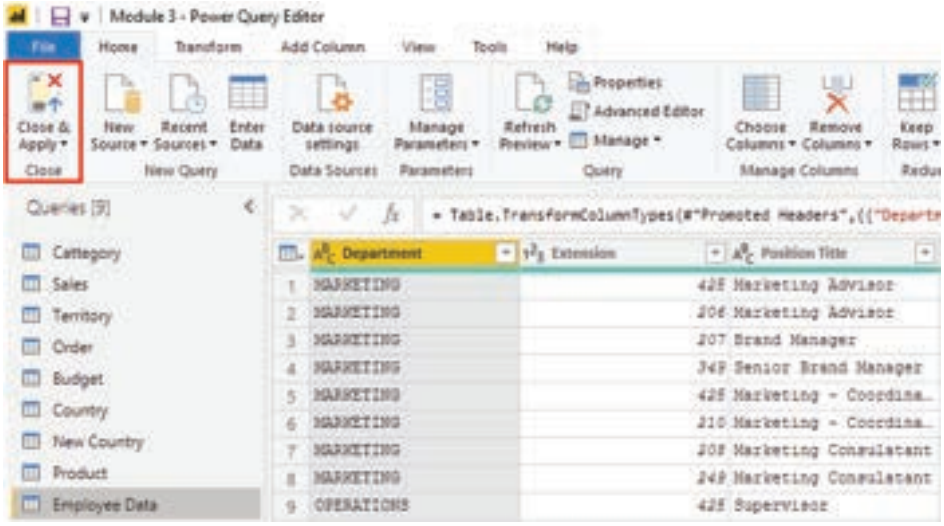


Figure 3.1.38.: Close and Apply step in Power Query.

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.

3.1.6. GET DATA FROM ONLINE SERVICES

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're 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, we've provided an example 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've 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 don't need to enter your full URL file path; you only need to load your site URL because, when you're 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.

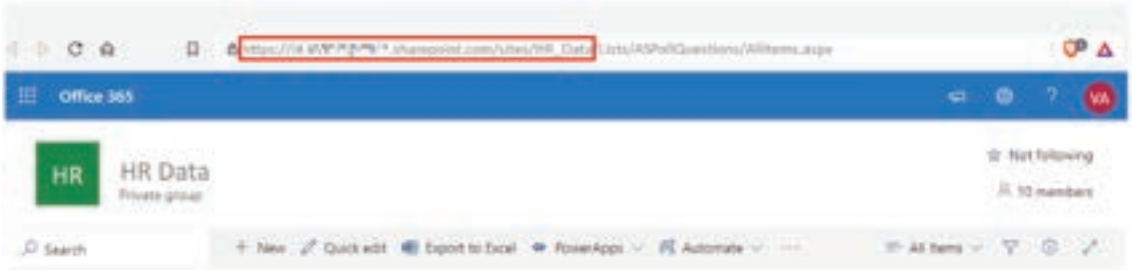


Figure 3.1.39.: SharePoint Online List URL.

After you've 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.

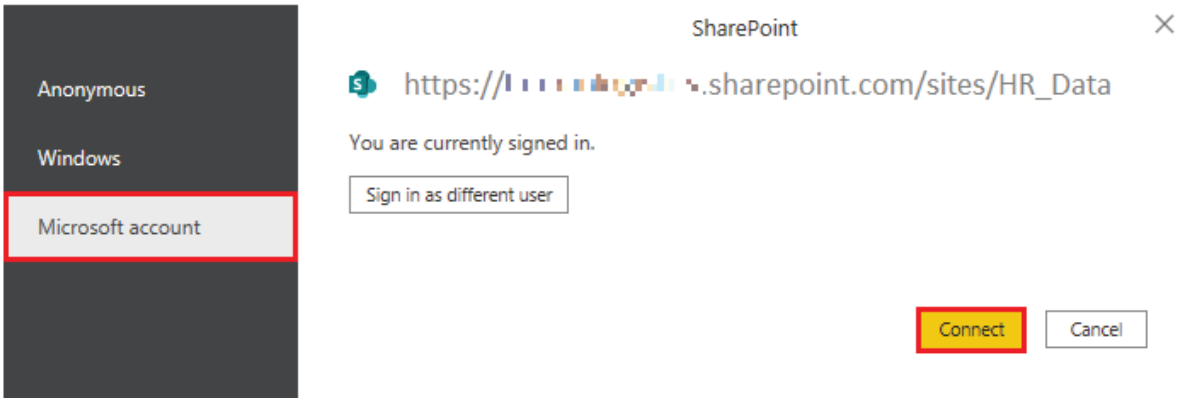


Figure 3.1.40.: Screenshot of the Authorization step to get access to SharePoint.

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 a Power BI model or launch the Power Query Editor to transform your data before loading it.

For this example, you select the Load option.

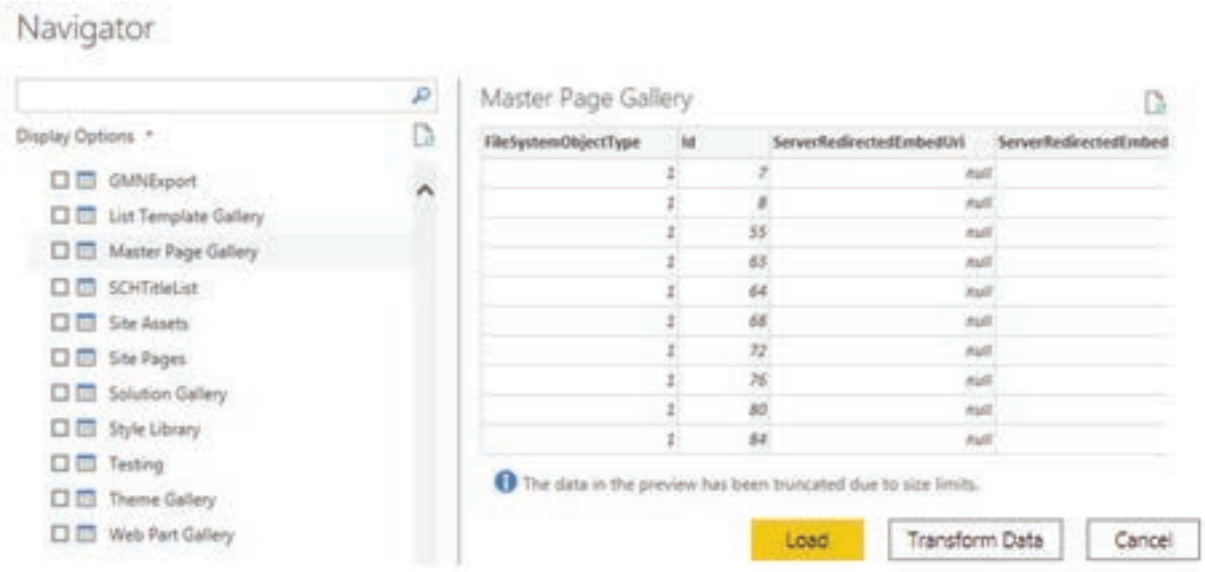


Figure 3.1.41.: Screenshot of the Navigator window view with available tables.

3.1.7. 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're building Power BI reports for the Sales department at Tailwind Traders, where importing the data isn't 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 aren't 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're 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're 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.

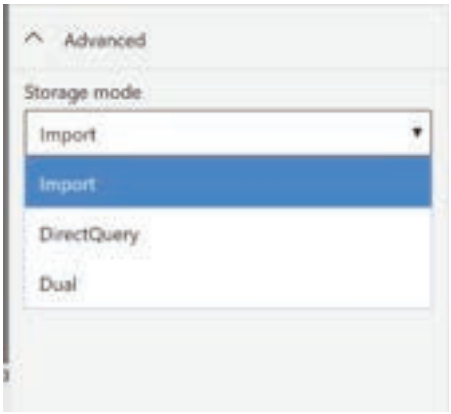


Figure 3.1.42:. Screenshot of the expanded storage mode list.

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. Data refreshes can be scheduled or on-demand. Import mode is the default for creating new Power BI reports.

DirectQuery mode:

The DirectQuery option is useful when you don't want to save local copies of your data because your data won't be cached. Instead, you can query the specific tables that you'll need by using native Power BI queries, and the required data will be retrieved from the underlying data source. Essentially, you're creating a direct connection to the data source. Using this model ensures that you're 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.

3.1.8. GET DATA FROM AZURE ANALYSIS SERVICES

Azure Analysis Services is a fully managed platform as a service (PaaS) that provides enterprise-grade data models in the cloud. You can use advanced mashup and modeling features to combine data from multiple data sources, define metrics, and secure your data in a single, trusted tabular semantic data model. The data model provides an easier and faster way for users to perform ad hoc data analysis using tools like 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 server is similar to getting data from SQL Server, in that you can:

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

Notable differences between Azure Analysis Services and SQL Server are:

- Analysis Services models have calculations already created.
- 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).

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're prompted for the server address and the database name with two options: Import and Connect live.

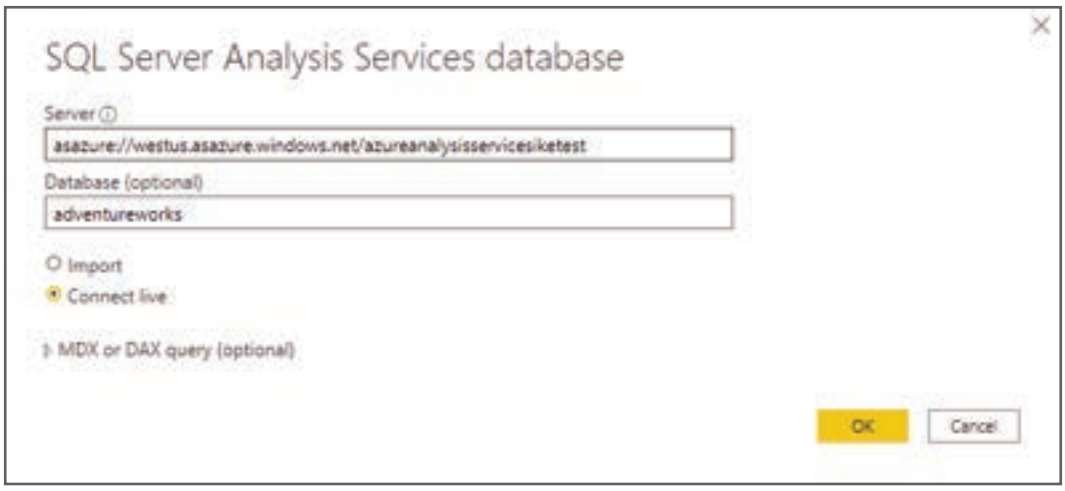


Figure 3.1.43.: SQL Server Analysis Services database dialog.

Connect live is an option for 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.

You'll 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. This approach simplifies your solution by keeping the data modeling and DAX measures in one place.

3.1.9. 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're 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.

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 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 include index creation, hardware upgrades, execution plan tuning, and data compression. These topics 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're 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 don't overwhelm Power BI computing resources.

You can use Power Query to load data into Power BI. Then use Power Query Editor to transform 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 doesn't 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're 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.

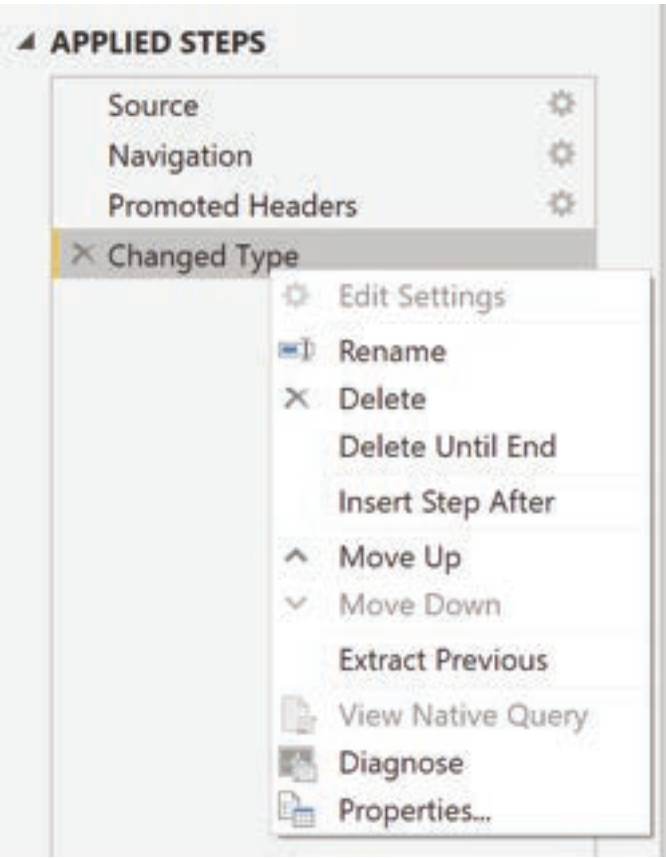


Figure 3.1.44.: Last applied step right-clicked to show the context menu.

If the View Native Query option isn't available (not displayed in bold type), then query folding isn't possible for this step, and you'll 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 aren't 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

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.

Query diagnostics:

Another tool that you can use to study query performance is query diagnostics. You can determine what bottlenecks may 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're ready to begin transforming your data or making other edits in Power Query Editor, select Start Diagnostics in the Session Diagnostics section. When you're finished, make sure that you select Stop Diagnostics.

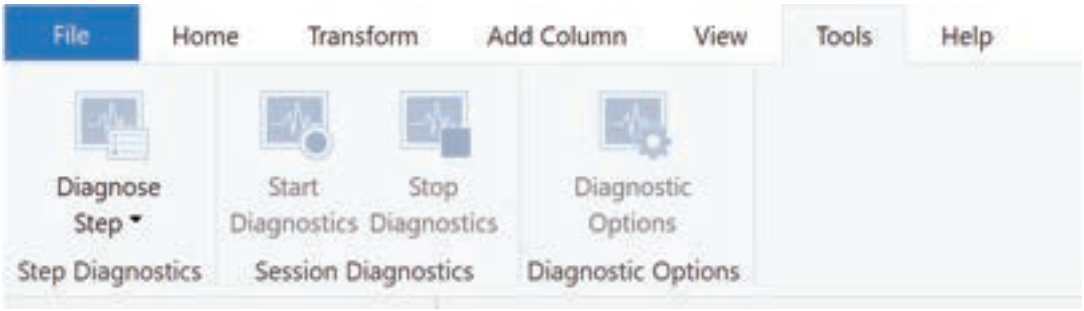


Figure 3.1.45.: Tools tab with session diagnostics options in the Power query Editor.

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.

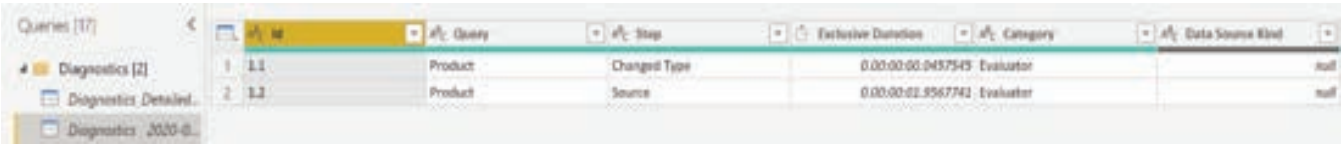


Figure 3.1.46.: Applying query diagnostics

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 aren't 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.

3.1.10. 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 often can't 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.

Figure 3.1.47.: Data import errors for query timeout.

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're 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're 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 even tells you the resolution. Perform the following steps to resolve the issue:

- Open your Excel workbook, and highlight the data that you want to import.
- Press the Ctrl-T keyboard shortcut. The first row will likely be your column headers.
- 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.

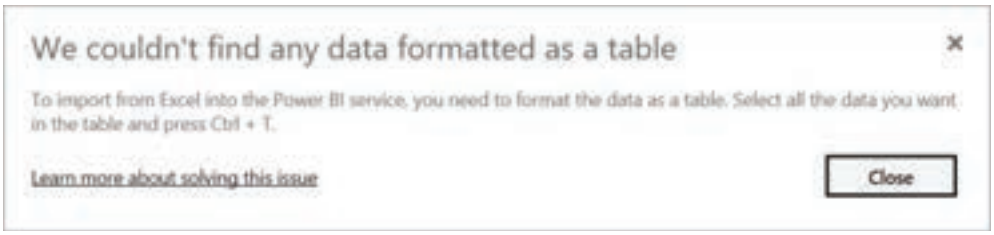


Figure 3.1.48.: Power B I Excel error. We couldn't find any data formatted as a table.

Couldn't find file:

While importing data from a file, you may get the "Couldn't find file" error.

Figure 3.1.49.: Could not find file error screen.

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.

- Open Power Query by selecting the Transform Data button in Power BI.
- Highlight the query that is creating the error.
- On the left, under Query Settings, select the gear icon next to Source.
- Change the file location to the new location.

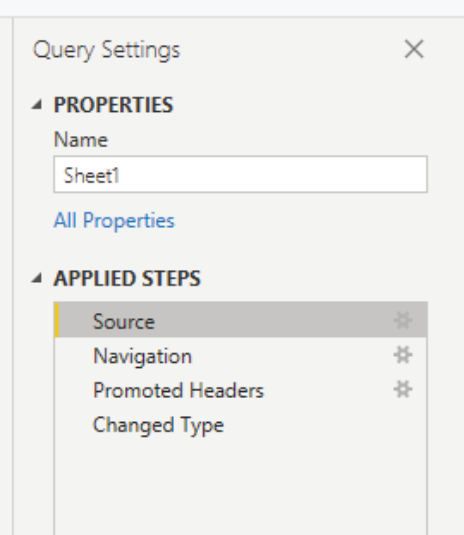


Figure 3.1.50.: File location settings pane.

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're 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.

SUMMARY

- Organizations frequently store data in flat files, including CSV, TXT, and Excel files. Power BI can import and analyze data from these files efficiently.
- Power BI allows direct connectivity to relational databases, enabling real-time data monitoring, trend analysis, and performance tracking.
- Dynamic reports empower users to customize data views by applying filters and selecting specific parameters, reducing the need for multiple static reports.
- Some organizations use NoSQL databases, which offer flexibility but differ from traditional relational databases. Power BI can retrieve data from these sources.
- Power BI can consolidate data from various online services like SharePoint, Google Analytics, and Dynamics 365 to generate comprehensive insights and reports.
- Power BI provides options for importing data into datasets or using DirectQuery to query data directly from the source, ensuring the most up-to-date information is accessible.
- Azure Analysis Services offers cloud-based data modeling capabilities, allowing users to combine data from multiple sources and perform ad hoc analysis with Power BI.
- The Performance Analyzer tool in Power BI assists in identifying and resolving performance-related problems while creating and running reports.
- Power BI may encounter data import errors due to various factors, including data source discrepancies and hardware issues. The document highlights common error messages and offers guidance on handling them.

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 What is the first step in creating reports in Power BI when dealing with data from various sources?
   a) Building reports
   b) Extracting data
   c) Cleaning data
   d) Publishing reports

2 Which type of file format is commonly used for flat files?
   a) PDF
   b) CSV
   c) JSON
   d) dHTML

3 What is the benefit of using cloud options like OneDrive or SharePoint for data storage in Power BI?
   a) Enhanced data security
   b) Real-time data synchronization
   c) Cost-effective data storage
   d) Improved data cleaning capabilities

4 How do you connect to a relational database in Power BI?
   a) Use the "Get data" feature and select the appropriate database option
   b) Use the "Create dynamic reports" feature
   c) Write an SQL query in Power BI
   d) Import a CSV file

5 Which of the following options is NOT a valid sign-in option when connecting to a relational database?
   a) Windows
   b) Database
   c) Gmail
   d) Microsoft account

6 What is the best practice when writing an SQL query to import data into Power BI?
   a) Use the wildcard character (*) in the SELECT statement
   b) Avoid using the WHERE clause
   c) Include unnecessary columns in the query
   d) Specify only the required columns and rows

7 What is the purpose of creating parameters in Power BI reports?
   a) To extract data from multiple sources
   b) To simplify the SQL query writing process
   c) To allow users to filter data based on their preferences
   d) To automate report publication

8 How can you create dynamic reports for multiple values using parameters in Power BI?
   a) By using the wildcard character (*) in SQL queries
   b) By disabling the "Enable load" option for Query1
   c) By connecting to an Excel worksheet with a table of values
   d) By using the "Transform Data" feature

9 What type of database is Cosmos DB?
- a) Relational database
- b) NoSQL database
- c) Document database
- d) Flat file database

10 How should you handle JSON data from a NoSQL database when importing it into Power BI?
- a) Import it directly without any transformation
- b) Normalize the data before loading it into Power BI
- c) Convert it into a CSV file format
- d) Use the "DirectQuery" option to connect to it

11 What is the purpose of connecting Power BI to external applications like SharePoint or OneDrive?
- a) To create more data sources
- b) To import data into Power BI
- c) To produce meaningful insights and reports
- d) To perform data transformations

12 In the provided scenario, why does Tailwind Traders want to establish a connection to SharePoint within Power BI?
- a) To import sales data into SharePoint
- b) To create a direct connection to sales data
- c) To access Power Query Editor
- d) To use Microsoft account credentials

13 Which category in Power BI Desktop should you select when connecting to data in an application like SharePoint Online List?
- a) Excel
- b) Online Services
- c) Database
- d) Web

14 What should you enter when prompted for your SharePoint URL in Power BI when connecting to an online list?
- a) Full file path
- b) Site URL only
- c) Username and password
- d) API key

15 Which storage mode allows you to query data in the data source directly without importing a copy into Power BI?
   a) Import
   b) Dual (Composite)
   c) DirectQuery
   d) Incremental

16 What is the benefit of using DirectQuery storage mode in Power BI?
   a) Faster data loading
   b) Improved data compression
   c) Always viewing the most recent data
   d) Access to Quick Insights

17 Which tool is used to address performance issues when running Power BI reports?
   a) Performance Analyzer
   b) Query Diagnostics
   c) Power Query Editor
   d) Data Transformer

18 What is the process by which Power Query Editor optimizes performance by tracking transformations as native queries?
   a) Query Tracking
   b) Data Optimization
   c) Query Folding
   d) Transformation Monitoring

19 What should you do if you encounter a "Query timeout expired" error in Power BI?
   a) Reduce the number of tables
   b) Increase data complexity
   c) Use DirectQuery exclusively
   d) Avoid using query folding

20 How can you resolve the "We couldn't find any data formatted as a table" error when importing data from Excel into Power BI?
   a) Highlight the data and press Ctrl-T
   b) Convert the data to JSON format
   c) Export data to CSV format
   d) Use Power Query Editor

Answers

1 *b. Extracting data*
2 *b. CSV*
3 *b. Real-time data synchronization*
4 *a. Use the "Get data" feature and select the appropriate database option*
5 *c. Gmail*
6 *d. Specify only the required columns and rows*
7 *c. To allow users to filter data based on their preferences*
8 *c. By connecting to an Excel worksheet with a table of values*
9 *b. NoSQL database*
10 *b. Normalize the data before loading it into Power BI*
11 *C) To produce meaningful insights and reports*
12 *B) To create a direct connection to sales data*
13 *B) Online Services*
14 *B) Site URL only*
15 *C) DirectQuery*
16 *C) Always viewing the most recent data*
17 *A) Performance Analyzer*
18 *C) Query Folding*
19 *A) Reduce the number of tables*
20 *A) Highlight the data and press Ctrl-T*

SELF-EXAMINATION QUESTIONS FOR PRACTICE:

1 What are the common file formats mentioned in the document that Power BI can work with?

2 In what situations might an organization prefer to use flat files like CSV or TXT for data storage and analysis?

3 Why is it advantageous for organizations to connect Power BI directly to relational databases instead of using flat files?

4 What types of insights can be gained by connecting Power BI to a sales database?

5 What is the primary benefit of creating dynamic reports in Power BI?

6 How do parameters allow users to interact with and customize data in Power BI reports?

7 What characterizes a NoSQL database, and how does it differ from a traditional relational database?

8 How does Power BI handle data retrieval from NoSQL databases, and what challenges might be associated with this process?

9 How can combining data from multiple online services enhance the insights and reports generated in Power BI?

CHAPTER 3

3.2. CLEAN, TRANSFORM, AND LOAD DATA IN POWER BI

Power Query has an incredible number of features that are dedicated to helping you clean and prepare your data for analysis. You'll learn how to simplify a complicated model, change data types, rename objects, and pivot data. You'll also learn how to profile columns so that you know which columns have the valuable data that you're seeking for deeper analytics.

LEARNING OBJECTIVES

- Resolve inconsistencies, unexpected or null values, and data quality issues.
- Apply user-friendly value replacements.
- 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.
- Edit M code in the Advanced Editor.

3.2.1. 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 a 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 that 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.
- Apply user-friendly value replacements.
- 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.
- Edit M code in the Advanced Editor.

3.2.2. SHAPE THE INITIAL DATA

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.

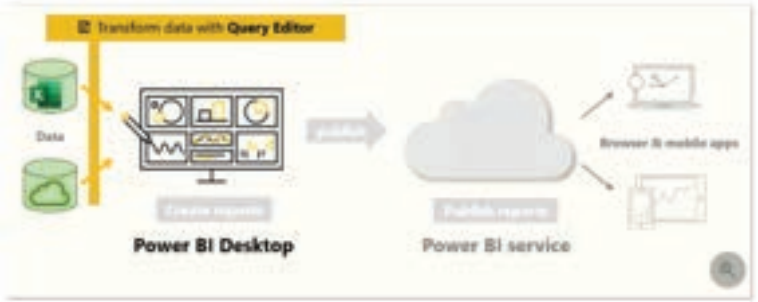


Figure 3.2.1.: Transform data with query editor

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.

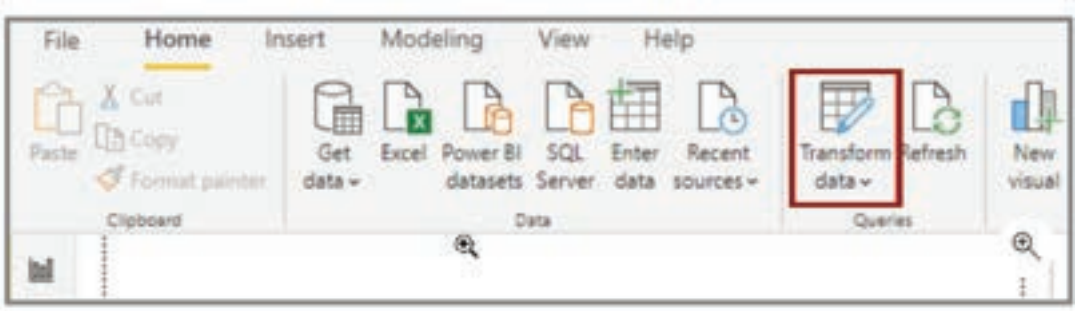


Figure 3.2.2.: Open power query editor

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.

Note:

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

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 csv file for SalesTarget (sample not provided) had a target categorized by products and a subcategory split by months, both of which are organized into columns.

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

Figure 3.2.3.: Original excel data

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

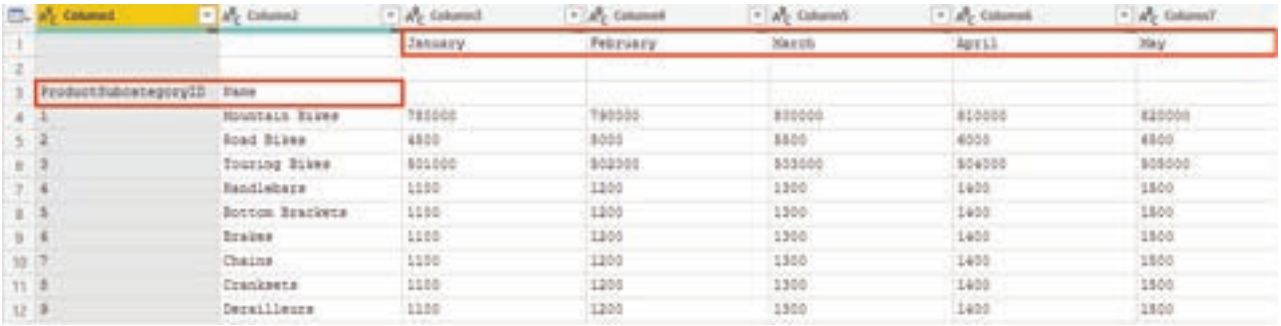


Figure 3.2.4.: Identify column headers and names

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.

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.

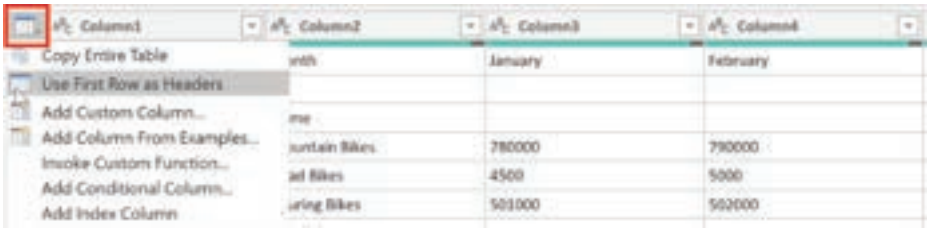


Figure 3.2.5.: Different options to use first row as headers

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

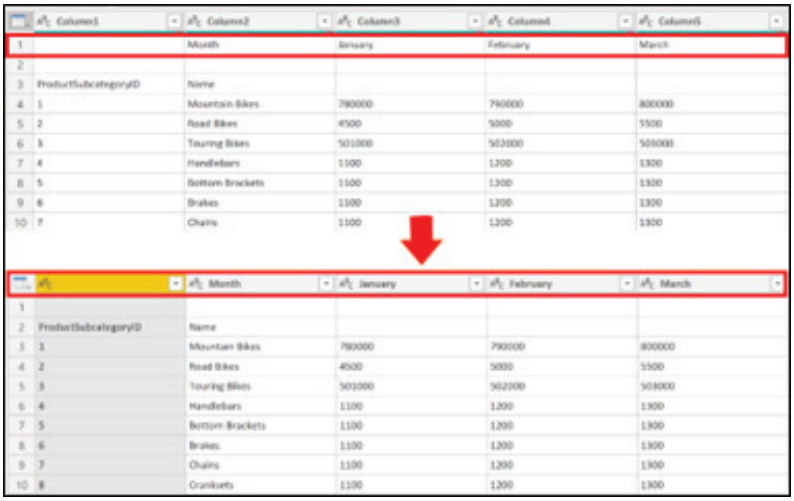


Figure 3.2.6.: Use first row as headers result

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 with the 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.

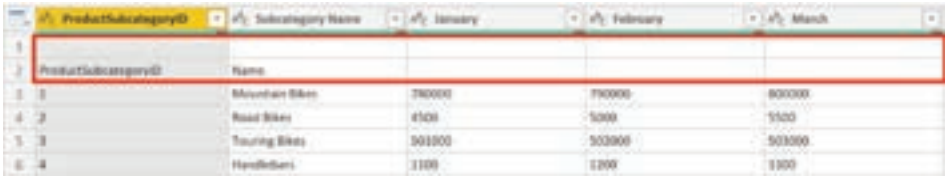


Figure 3.2.7.: Remove top rows

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

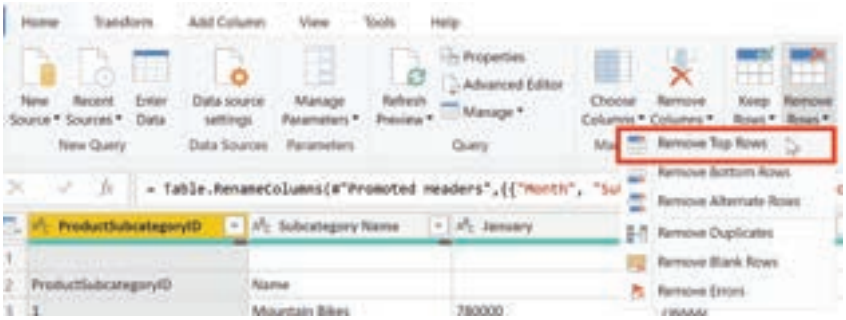


Figure 3.2.8.: Remove top rows feature

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.

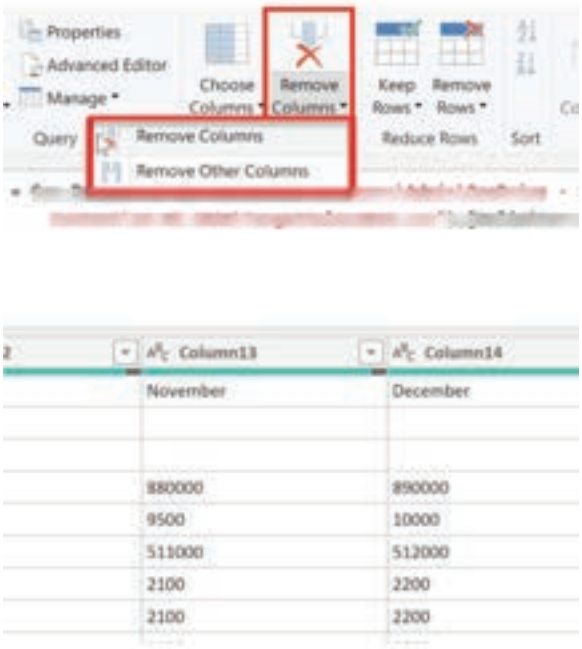


Figure 3.2.9.: Remove columns feature

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.

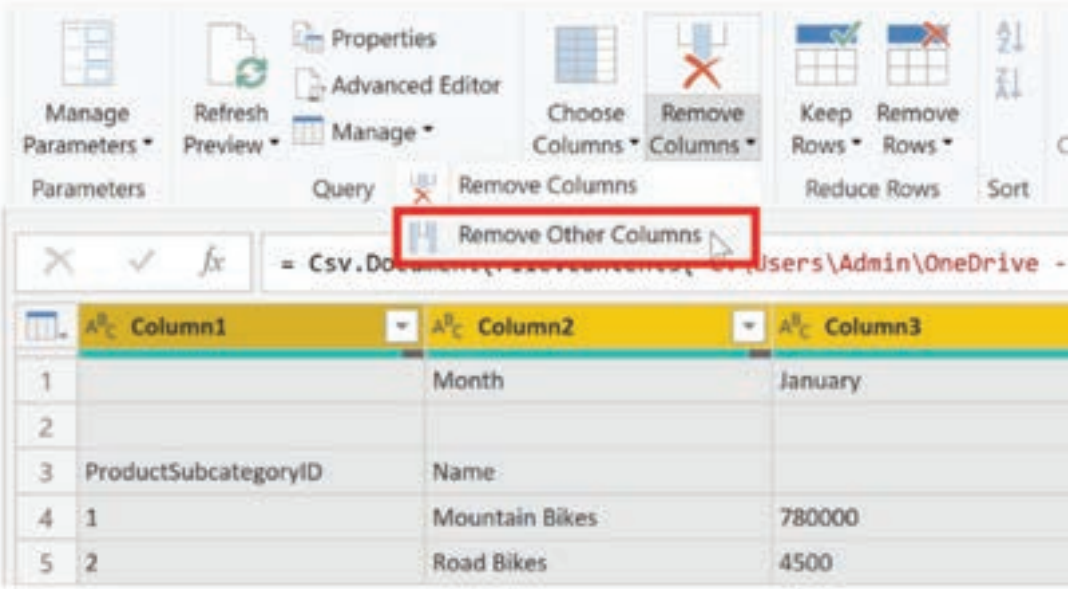


Figure 3.2.10.: Remove other columns feature

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
Novembe	$	8,014	$	9,299
Decembe	$	19,470	$	14,082

Figure 3.2.11.: Excel data that needs to be unpivoted

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

Figure 3.2.12.: Original Power Query data

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

Figure 3.2.13.: Unpivot results in Power Query

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.

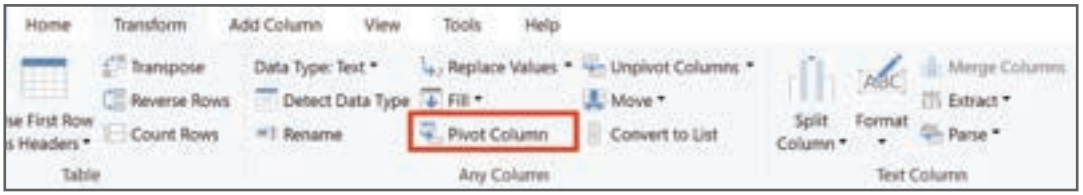


Figure 3.2.14.: Pivot Column

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.

Aggregate value function

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

Figure 3.2.15.: Pivot column feature changes how 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.

3.2.3. SIMPLIFY THE DATA STRUCTURE

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.

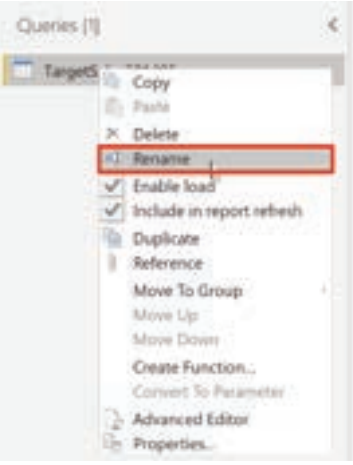


Figure 3.2.16.: Rename a query

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.

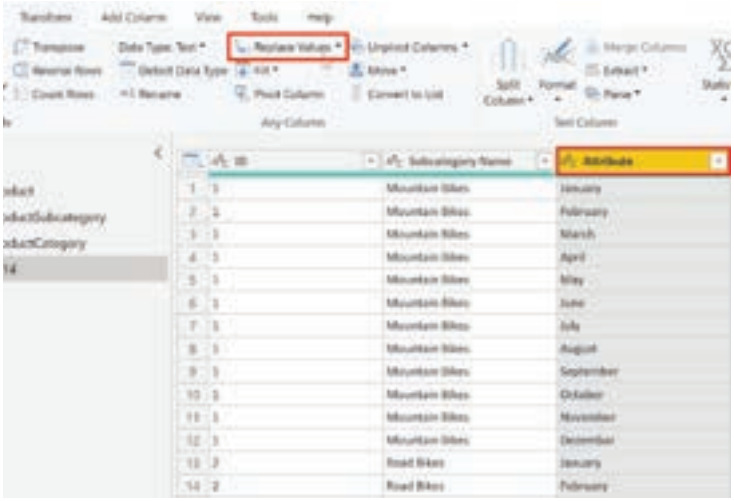


Figure 3.2.17.: Replace value feature in Power Query Editor

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.

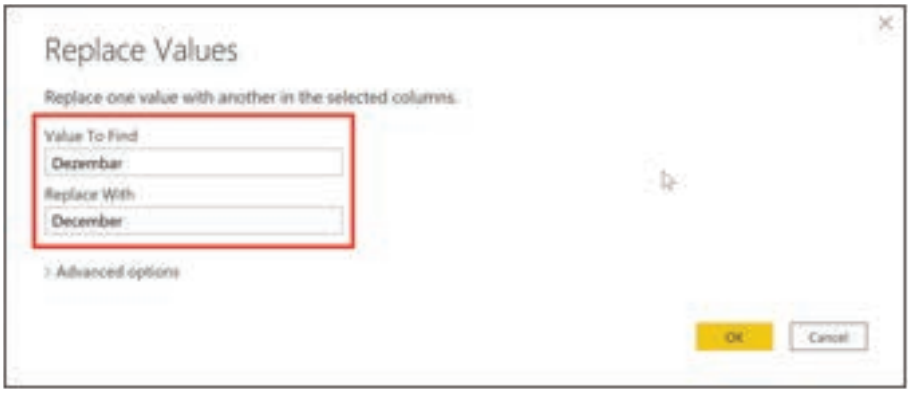


Figure 3.2.18.: Replace one value with another in the value to find box

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, the averages 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.

Figure 3.2.19.: Replace null value 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.

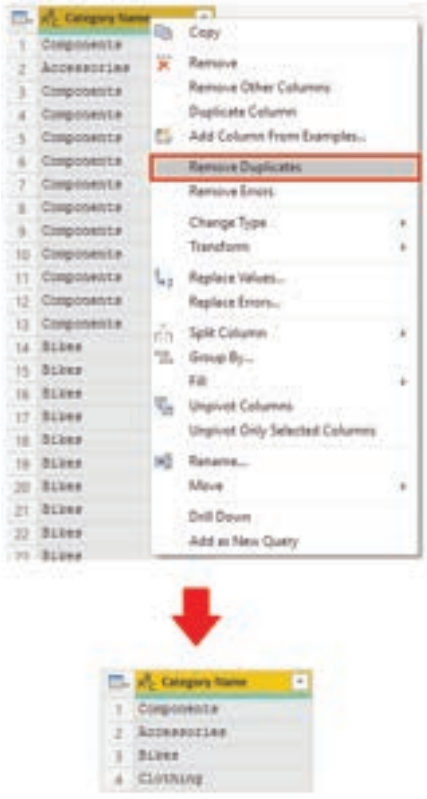


Figure 3.2.20. Remove duplicates feature

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, prefixes, 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.

3.2.4. 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 doesn't detect the correct data type. Where incorrect data types occur, you'll experience performance issues.

You have a higher chance of getting data type errors when you're 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're 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 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

Figure 3.2.21.: Column OrderDate data type set as text

Implications of incorrect data types:

The following information provides insight into problems that can arise when Power BI doesn't 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'll get the following error stating that the OrderDate column data type isn't Date, which is required in time-based calculations.

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

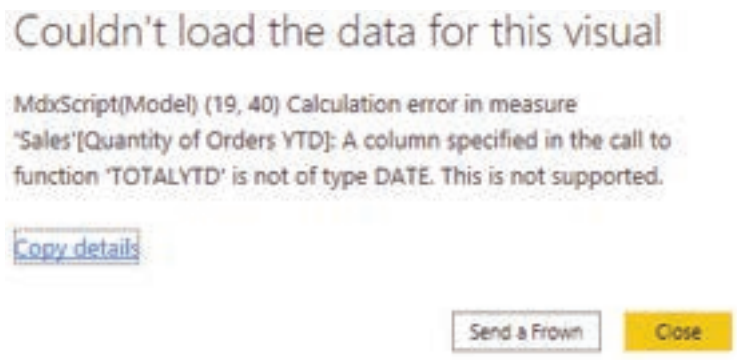


Figure 3.2.22.: Error for time based calculated measure

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 a yearly, monthly, or weekly basis. The following screenshot shows that the SalesDate field isn't recognized as type Date and will only be presented as a list of dates in the Table visual. However, It's 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 documentation.

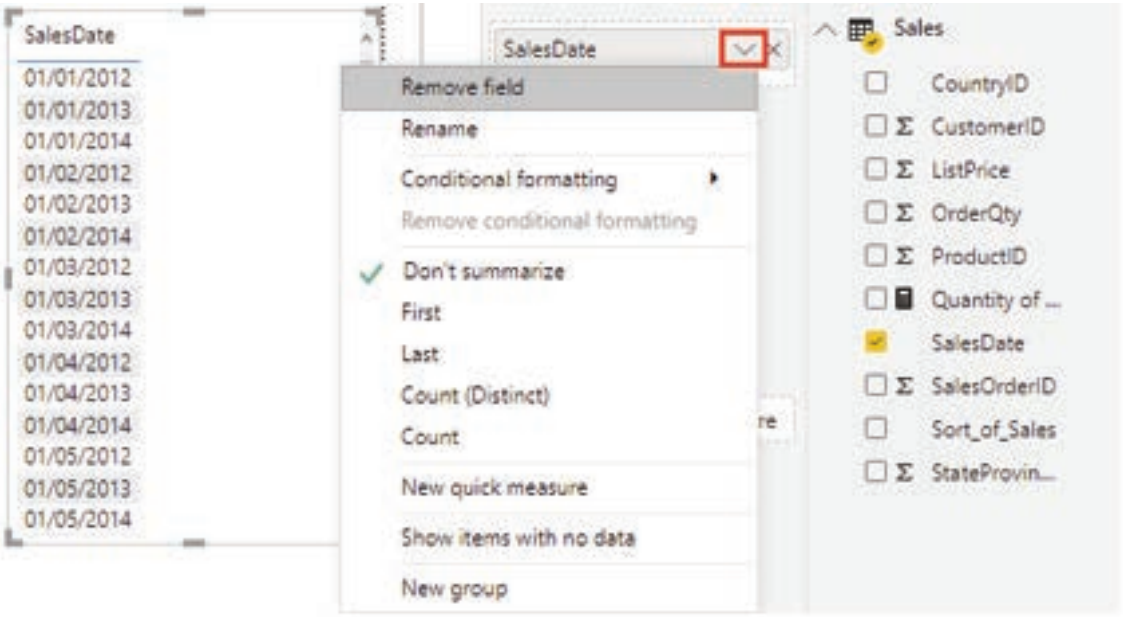


Figure 3.2.23.: Additions SalesDate options

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.

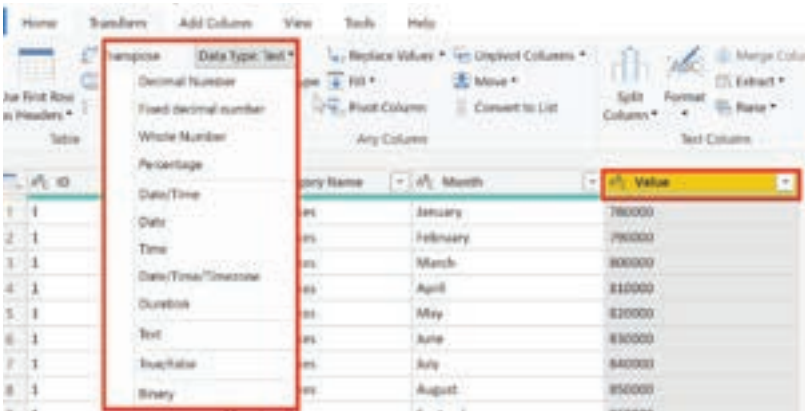


Figure 3.2.24.: Select data type under transform ribbon

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

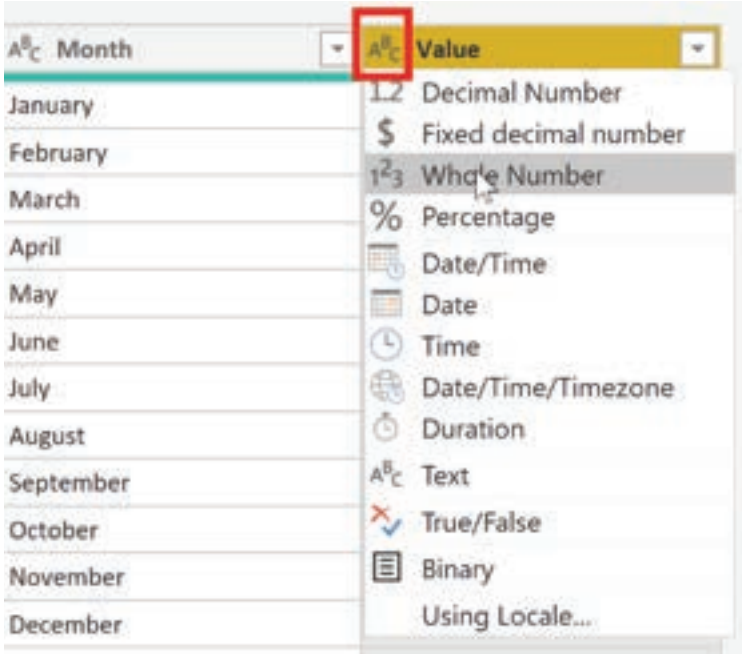


Figure 3.2.25.: Select data type from list

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.

3.2.5. 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're 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.

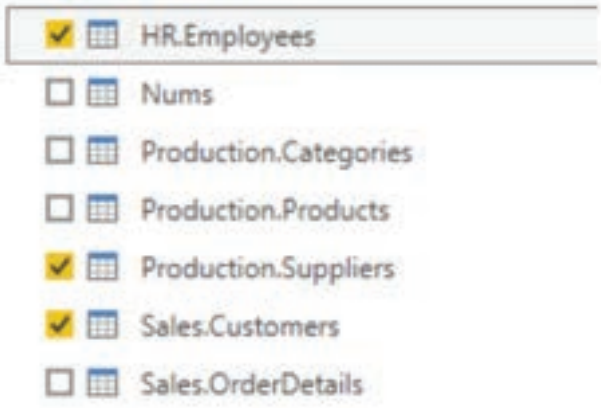


Figure 3.2.26.: Screenshot of choosing tables in Power Query Editor

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'll 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'll end up with 400 rows. When you merge queries, you'll 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'll 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.

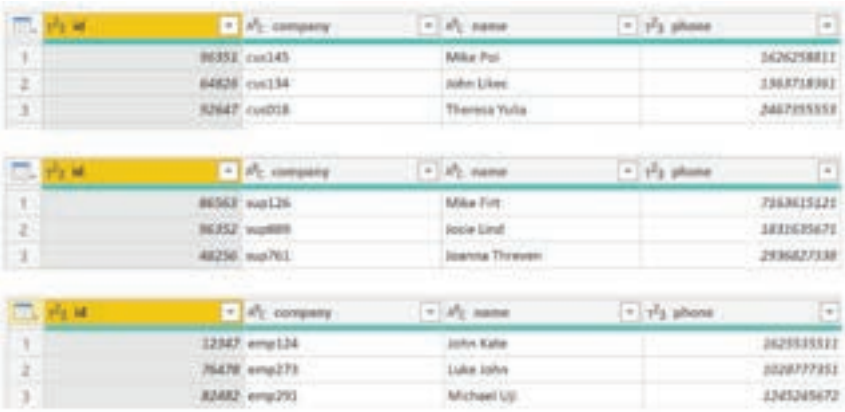


Figure 3.2.27.: Reformatting for appending

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.

Figure 3.2.28.: Append Queries as New in Power Query Editor

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

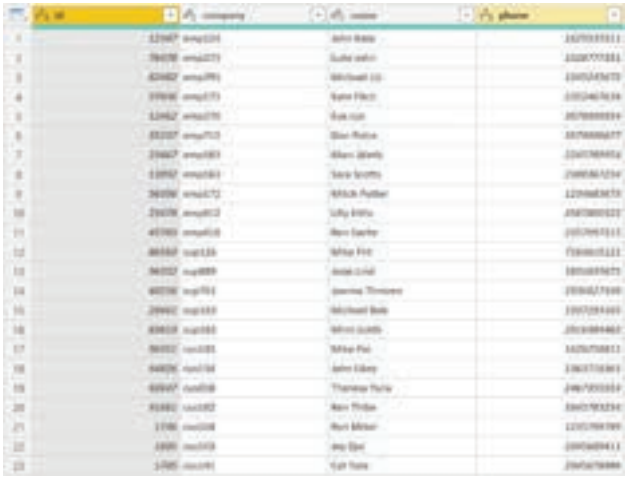


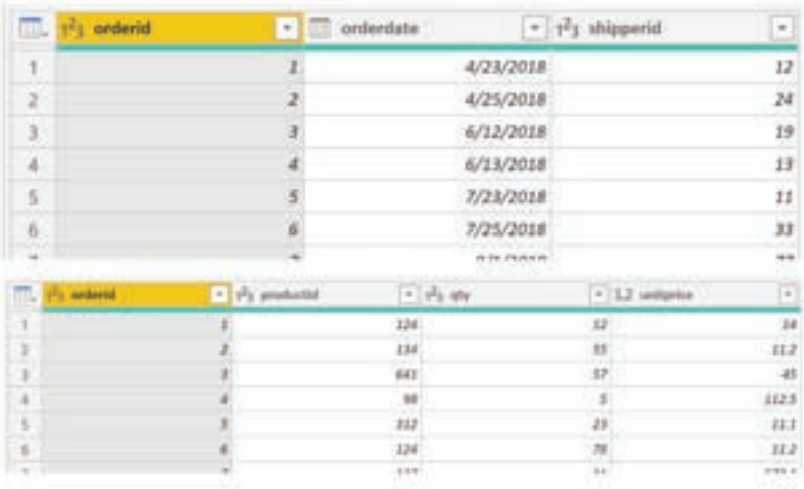
Figure 3.2.29.: Append as new final

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're 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

Figure 3.2.30.: Order and Order Details tables to be merged

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 ordered.

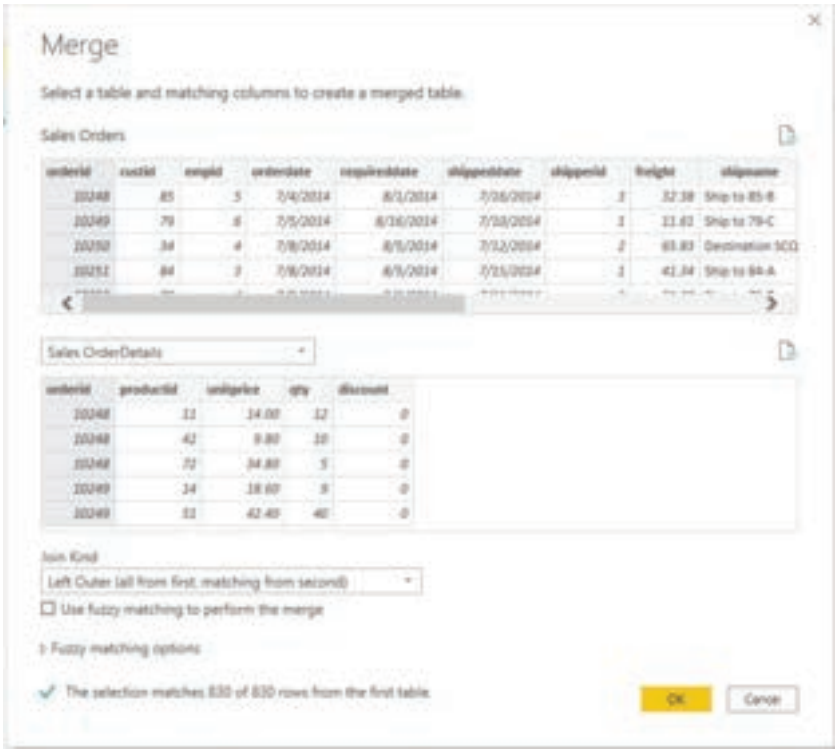


Figure 3.2.31.: Merge queries window

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'll choose to use a Left Outer join. Select OK, which will route you to a new window where you can view your merged query.

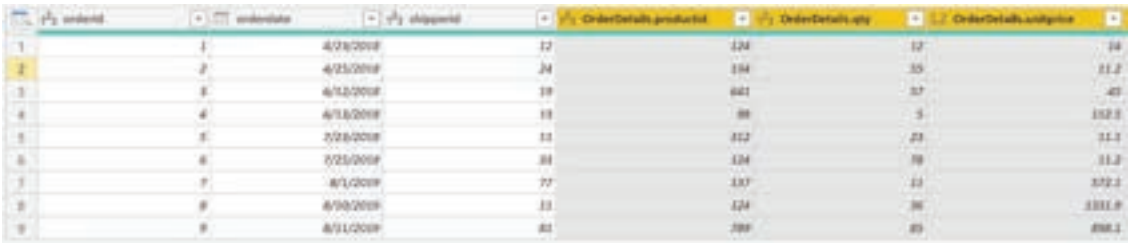


Figure 3.2.32.: Merged Queries final view

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.

3.2.6. PROFILE DATA IN POWER BI

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.

Figure 3.2.33.: Example data structure and ribbon bar

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.

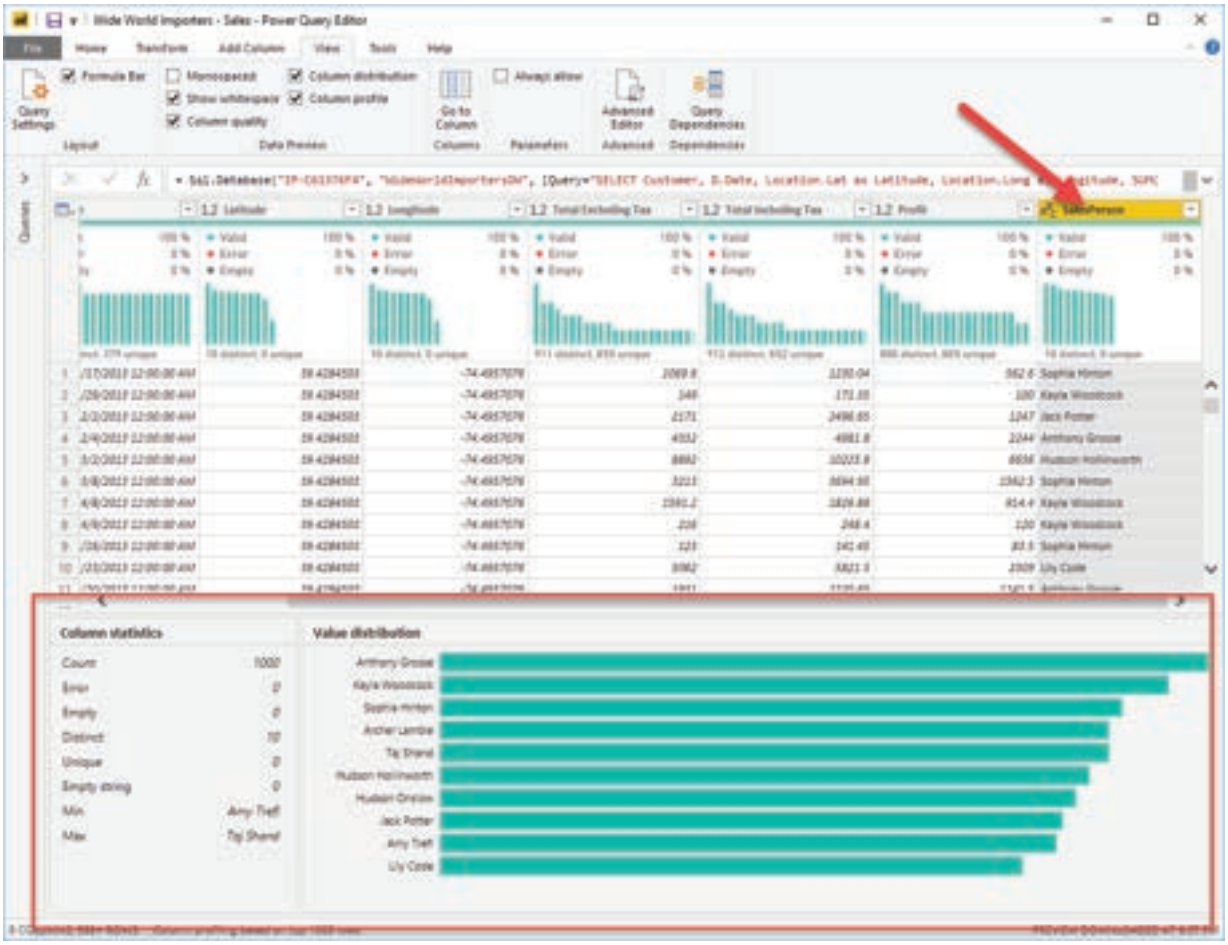


Figure 3.2.34.: Anomalies and data statistics for a column of data.

Note:

By default, Power Query examines the first 1000 rows of your data set. To change this, select the profiling status in the status bar and select Column profiling based on entire data set.

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 the different 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 only appear once.

Column profile gives you a more in-depth look into the statistics within the columns for the first 1,000 rows of data. 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, 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 distinct value in that specific column. When looking at the graph in the previous image, notice that the value distribution indicates that "Anthony Gross" appears the greatest number of times within the SalesPerson column and that "Lily Code" appears the least number 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 number 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, find data anomalies, and view 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.

3.2.7. USE ADVANCED EDITOR TO MODIFY M CODE

Each time you shape data in Power Query, you create a step in the Power Query process. Those steps can be reordered, deleted, and modified where it makes sense. Each cleaning step that you made was likely created by using the graphical interface, but Power Query uses the M language behind the scenes. The combined steps are available to read by using the Power Query Advanced Editor. The M language is always available to be read and modified directly. It is not required that you use M code to take advantage of Power Query. You will rarely need to write M code, but it can still prove useful. Because each step in Power Query is written in M code, even if the UI created it for you, you can use those steps to learn M code and customize it to suit your needs.

After creating steps to clean data, select the View ribbon of Power Query and then select Advanced Editor.

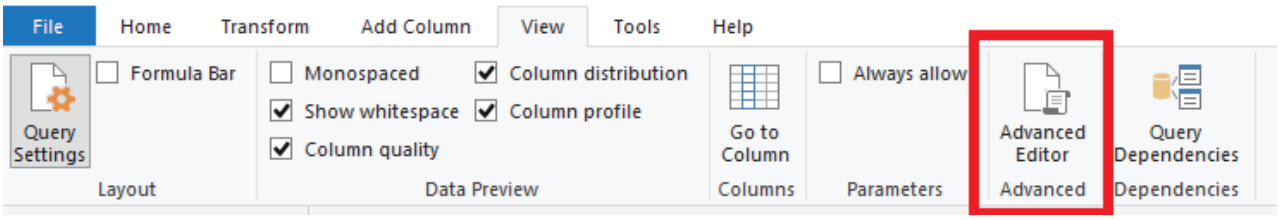


Figure 3.2.35.: View ribbon, advanced editor button

The following screen should appear.

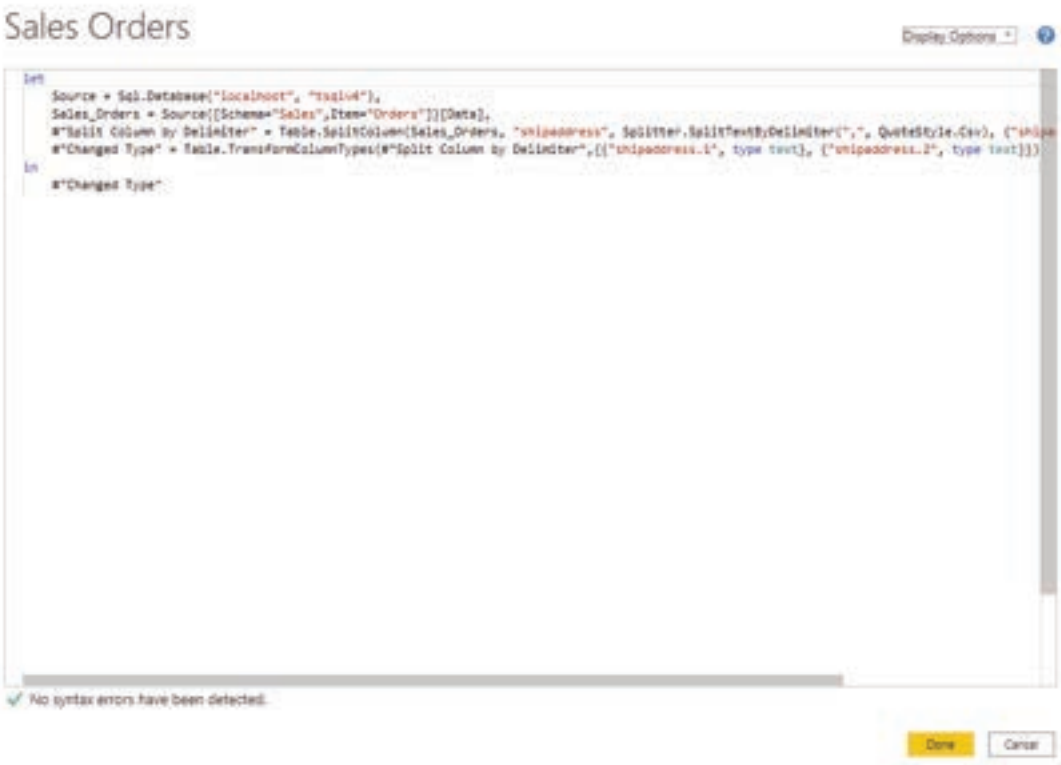


Figure 3.2.36.: M language in advanced editor

Each Power Query step will roughly align with one or two lines of M code. You don't have to be an expert in M code to be able to read it. You can even experiment with changing it. For instance, if you need to change the name of a database, you could do it right in the code and then select Done.

You might notice that M code is written top-down. Later steps in the process can refer to previous steps by the variable name to the left of the equal sign. Be careful about reordering these steps because it could ruin the statement dependencies. Write to a query formula step by using the in statement. Generally, the last query step is used as the in final data set result

SUMMARY

- Power Query Editor allows you to transform imported data by renaming columns, changing data types, removing unnecessary rows, and more. Data should be shaped to meet specific needs and report requirements.

- When importing data from multiple sources, you can use Power Query Editor to change table and column names for consistency and user-friendliness. This step simplifies the data structure.

- It's important to review and correct data types, especially when dealing with flat files where manual entry errors may occur. Incorrect data types can lead to performance issues.

- Combining queries allows you to merge or append tables when dealing with complex data models, similar roles, or when you want to use columns from different tables in a custom column.

- Profiling data involves studying data nuances, identifying anomalies, and examining data statistics to shape and organize it for easier interaction and report development.

- Power Query steps are created in the M language, even if generated through the graphical interface. The Advanced Editor allows you to read and modify M code, providing customization options for advanced users, although it's not always necessary.

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 What is one of the challenges that can make imported data unprepared for analysis in Power BI?
   a) Inconsistent column headers
   b) Excessive data volume
   c) Lack of data visualization
   d) Inaccurate data source

2 What advantage does clean data offer in Power BI?
   a) It reduces data import time.
   b) It simplifies report design.
   c) It eliminates the need for aggregations.
   d) It produces more accurate results.

3 How can you promote the first row of data as column headers in Power Query Editor?
   a) Select the "Use First Row as Headers" option on the Home tab.
   b) Manually type the headers in the Query Settings pane.
   c) Right-click on the header row and choose "Promote Headers."
   d) It is not possible to promote headers in Power Query Editor.

4 What is the purpose of renaming columns in Power Query Editor?
   a) To hide columns from view
   b) To enhance data security
   c) To make column names more descriptive
   d) To create new calculated columns

5 How can you remove top rows in Power Query Editor?
   a) Right-click on the rows to be removed and select "Delete."
   b) Select "Remove Rows > Remove Top Rows" on the Home tab.
   c) Manually delete the rows in the data source file.
   d) Use the "Filter Rows" option to exclude top rows.

6 When is it advisable to remove unnecessary columns in Power Query?
   a) After building all visualizations
   b) Before establishing relationships between tables
   c) Only when there is insufficient storage space
   d) As the final step in data preparation

7 What is the purpose of unpivoting data in Power Query?
   a) To add more columns to the dataset
   b) To create calculated columns
   c) To simplify the data structure
   d) To remove null values

8 When should you use the Pivot Column feature in Power Query?
   a) To add new columns to the dataset
   b) To summarize data by aggregating values
   c) To remove duplicate rows
   d) To change column data types

9 Where can you find a record of all the steps taken to shape your data in Power Query Editor?
   a) In the Power Query Editor ribbon
   b) In the Query Settings pane
   c) In the Transform tab
   d) In the Queries pane

10 What is the primary purpose of renaming queries in Power Query Editor?
   a) To hide queries from view
   b) To make queries more visually appealing
   c) To improve user-friendliness
   d) To reduce query execution time

11 How can you replace values in a selected column using Power Query Editor?
   a) Right-click on the values and select "Replace."
   b) Select "Replace Values" on the Home tab.
   c) Use the "Find and Replace" tool in Excel.
   d) It is not possible to replace values in Power Query Editor.

12 What is the purpose of replacing null values in Power Query Editor?
   a) To increase data storage capacity
   b) To improve data visualization
   c) To maintain data consistency
   d) To speed up data import

13 How can you remove duplicate values from a column in Power Query Editor?
   a) Right-click on the duplicates and select "Delete."
   b) Use the "Remove Duplicates" feature on the Transform tab.
   c) Manually edit the column in the source file.
   d) It is not possible to remove duplicates in Power Query Editor.

14 What is a best practice for naming tables, columns, and values in Power BI?
   a) Use excessively short abbreviations
   b) Avoid using prefixes or suffixes
   c) Use underscores instead of spaces
   d) Use common terminology within your organization

15 What should you consider when replacing values in Power Query Editor?
   a) Use acronyms to save space
   b) Ensure values are as short as possible
   c) Think about how values will appear in reports
   d) Replace values in the source data file

16 What can be a consequence of having incorrect data types in Power BI?
   a) Improved performance in calculations
   b) Easier creation of date hierarchies
   c) Inability to create certain calculations and relationships
   d) Enhanced data visualization options

17 When should you change the data type of a column in Power Query Editor?
   a) After loading the data into a Power BI data model
   b) Before evaluating the column data types
   c) In the Power BI Desktop Report view
   d) After creating a calculated measure

18 What is the primary purpose of appending queries in Power BI?
   a) Combining rows of data from one table to another
   b) Merging tables based on a common column
   c) Changing the data type of a column
   d) Creating calculated measures

19 How can you identify data anomalies and statistics in Power Query Editor?
   a) By using the Query Dependencies feature
   b) By selecting the Data Profiling option
   c) By examining the Model tab in Power BI Desktop
   d) By modifying M code in the Advanced Editor

20 What is the purpose of the Advanced Editor in Power Query?
   a) To reorder or delete data cleaning steps
   b) To create calculated measures
   c) To write and modify M code directly
   d) To create relationships between tables

Answers

1 *A) Inconsistent column headers*
2 *D) It produces more accurate results.*
3 *A) Select the "Use First Row as Headers" option on the Home tab.*
4 *C) To make column names more descriptive*
5 *B) Select "Remove Rows > Remove Top Rows" on the Home tab.*
6 *B) Before establishing relationships between tables*
7 *C) To simplify the data structure*
8 *B) To summarize data by aggregating values*
9 *B) In the Query Settings pane*
10 *C) To improve user-friendliness*
11 *B) Select "Replace Values" on the Home tab.*
12 *C) To maintain data consistency*
13 *B) Use the "Remove Duplicates" feature on the Transform tab.*
14 *D) Use common terminology within your organization*
15 *C) Think about how values will appear in reports*

16 *C) Inability to create certain calculations and relationships*
17 *B) Before evaluating the column data types*
18 *A) Combining rows of data from one table to another*
19 *B) By selecting the Data Profiling option*
20 *C) To write and modify M code directly*

SELF-EXAMINATION QUESTIONS FOR PRACTICE:

1 What is the primary purpose of using Power Query Editor in Power BI Desktop?

2 Why is it essential to shape data during the data preparation process for reports?

3 In what situations might you need to change column names in Power Query Editor?

4 How can incorrect data types in columns impact the performance of your Power BI model?

5 When is it advisable to evaluate and modify column data types in Power Query Editor?

6 What benefits does combining multiple tables into a single table offer in data modeling?

7 Why is data profiling important, and what kind of insights can it provide?

8 What role does the M code play in Power Query transformations, and when might you use the Advanced Editor to modify it?

9 How does simplifying the data structure contribute to a more user-friendly data model in Power BI?

CHAPTER 4
DATA VISUALIZATIONS AND DATA EXTRACTION

4.1. USE VISUALS IN POWER BI

Create and customize visuals to present data in compelling and insightful ways.

LEARNING OBJECTIVES

- Explore Power BI visuals
- Create visuals

4.1.1. INTRODUCTION TO VISUALS IN POWER BI

Visuals allow you to present data in a compelling and insightful way, and help you show the important components of it. Power BI has many compelling visuals and many more that are released frequently.

This unit begins with the mainstays of visualizations, the simple visuals that everyone's familiar with, to make sure that you know the particulars of them. The rest of the module will provide more advanced, or at least less common, details to enhance your report-creating knowledge.

Figure 4.1.1.: Screenshot of an example report with 4 common visuals.

Visualizing data is one of the core parts and basic building blocks of Power BI. Creating visuals is one of the most effective ways to find and share your insights.

You'll discover a wide variety of visualizations in Power BI, which offers features such as simple bar charts, pie charts, maps, and more esoteric offerings like waterfalls, funnels, and gauges. Power BI Desktop also offers extensive page formatting tools, such as shapes and images, that help bring your report to life.

4.1.2. CREATE AND CUSTOMIZE SIMPLE VISUALIZATIONS

This unit explains how to create new bar charts, pie charts, and tree maps, and how to customize these elements to suit your reports.

Two ways to create a new visualization in Power BI Desktop are:

- Drag field names from the Fields pane and then drop them on the report canvas. By default, your visualization appears as a table of data.

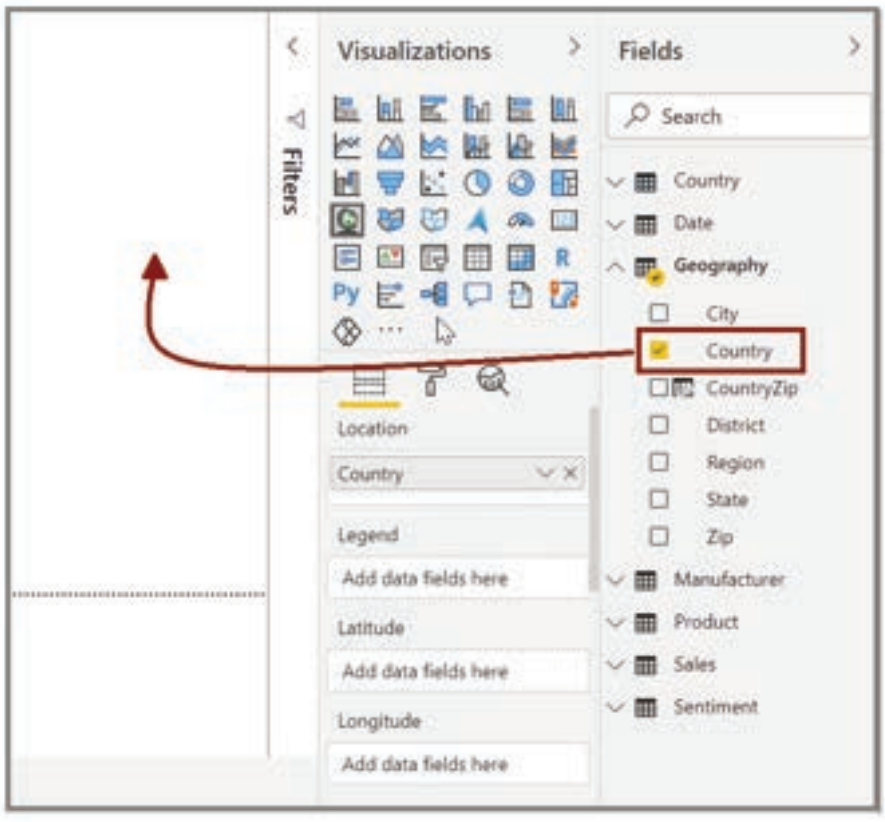


Figure 4.1.2.: Drag the "Country" field onto the canvas to create a visual.

In the Visualizations pane, select the type of visualization that you want to create. With this method, the default visual is a blank placeholder that resembles the type of visual that you selected.

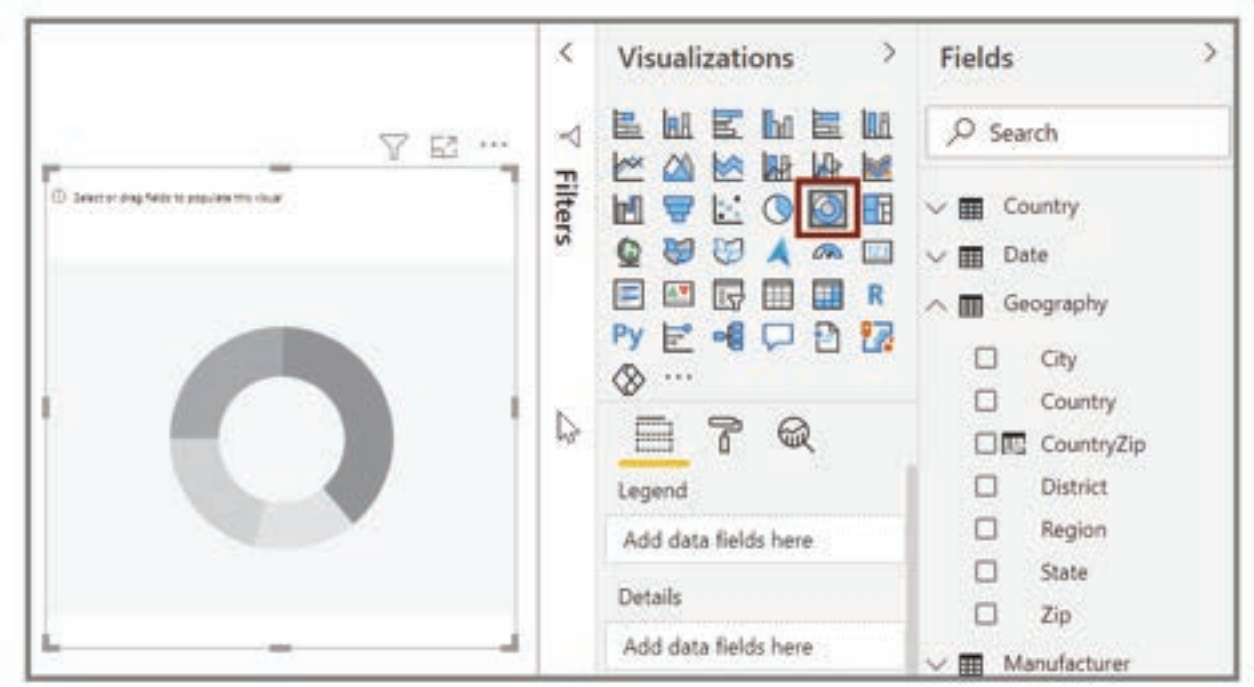


Figure 4.1.3.: Screenshot of a visual selected on the Visualizations pane.

After you have created your graph, map, or chart, you can begin dragging data fields onto the bottom portion of the Visualization pane to build and organize your visual. The available fields will change based on the type of visualization that you selected. As you drag and drop data fields, your visualization will automatically update to reflect changes.

You can resize your visual by selecting it and then dragging the handles in or out. You can also move your visualization anywhere on the canvas by selecting and then dragging it to where you want it. If you want to convert between different types of visuals, select the visual that you want to change and select a different visual from the Visualization pane. Power BI attempts to convert your selected fields to the new visual type as closely as possible.

As you hover over parts of your visuals, you'll receive a tooltip that contains details about that segment, such as labels and total value.

Select the paintbrush icon on the Visualizations pane to make cosmetic changes to your visual. Examples of cosmetic changes include background, alignment, title text, and data colors.

The available options for cosmetic changes to your visual vary depending on the type of visual that you've selected.

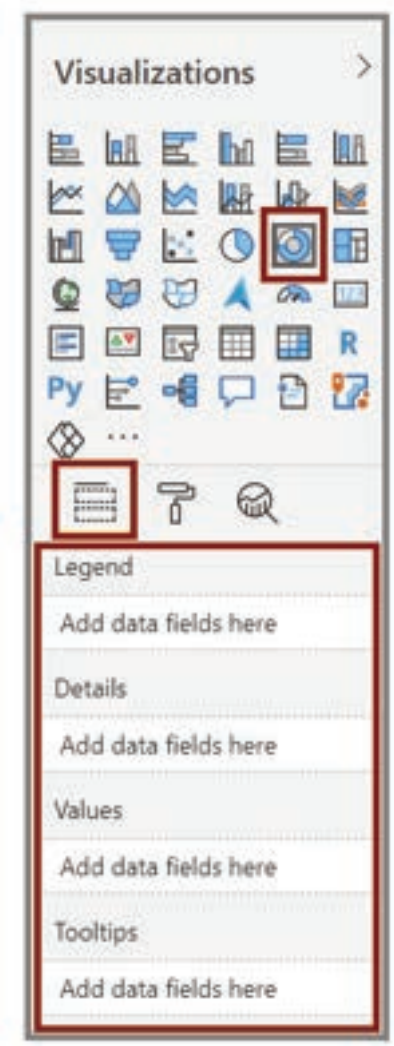


Figure 4.1.4.: Screenshot of the Visualizations options.

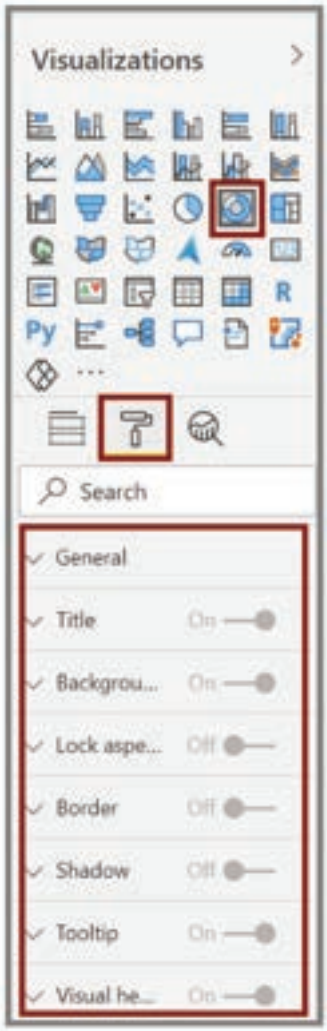


Figure 4.1.5.: The Visualizations format pane.

Note:

Generally, visuals are used to compare two or more different values. However, sometimes when you are building reports, you might want to track a single metric over time. For more information, see Radial gauge charts in Power BI.

Create combination charts:

Combination charts are an effective way to visualize multiple measures that have different scales in a single visualization.

You might want to visualize two measures with different scales, such as revenue and units. Use a combination chart to show a line and a bar with different axis scales. Power BI supports many different types of combination charts by default. To get the image below click on the line and clustered column chart in the Visualizations pane

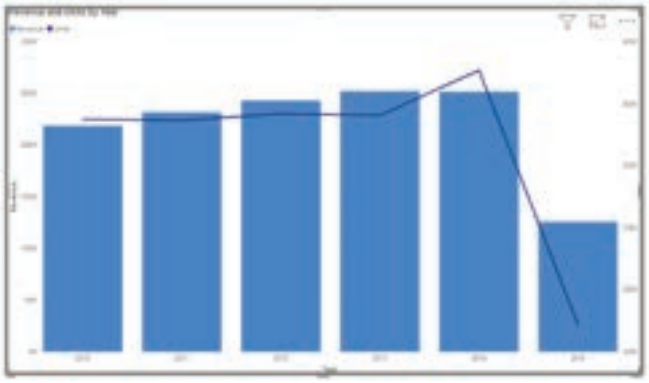


Figure 4.1.6.: Image of a line and bar chart combined in the same visual.

You can split each column by category by dragging a category into the Column Series field. When you do so, each bar is proportionately colored based on the values within each category.

4.1.3. CREATE SLICERS

Slicers are one of the most powerful types of visualizations, particularly as part of a busy report. A slicer is an on-canvas visual filter that allows report users to segment the data by a specific value. Examples of filters include by year or by geographical location.

To add a slicer to your report, select Slicer from the Visualizations pane.

Figure 4.1.7.: Image of the Slicer button on the Visualizations pane.

Drag the field by which you want to slice and drop it to the top of the slicer placeholder. The visualization turns into a list of elements with check boxes. These elements are your filters. Select the box next to the one that you want to segment, and Power BI will filter, or slice, all other visuals on the same report page.

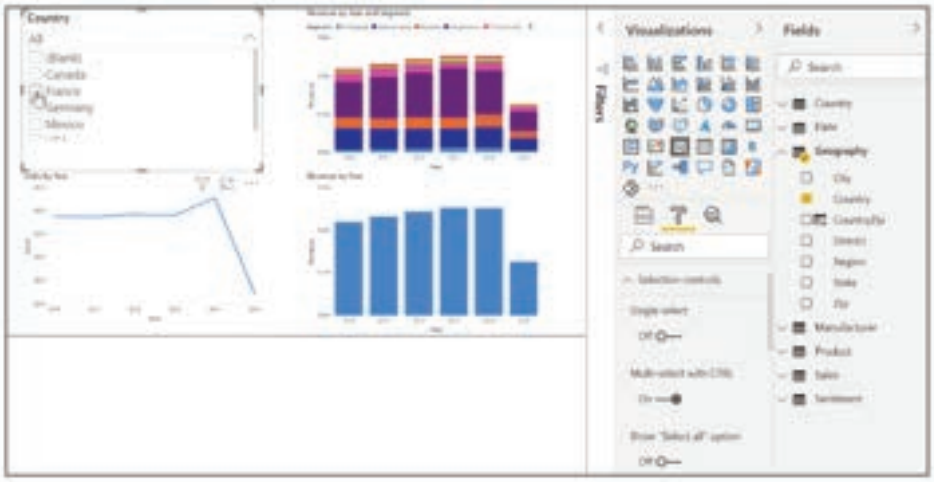


Figure 4.1.8.: Image of a field dragged onto the slicer placeholder.

A few different options are available to help you format your slicer. You can set it to accept multiple inputs at once, or you can use the Single Select mode to use one at a time. You can also add a Select All option to your slicer elements, which is helpful when you have a long list. Change the orientation of your slicer from the vertical default to horizontal, and it becomes a selection bar rather than a checklist.

Figure 4.1.9.: Image of of slicer formatting options.

When you have multiple visualizations on the same report page, Power BI Desktop lets you control how interactions flow between visuals.

4.1.4. MAP VISUALIZATIONS

Power BI has two different types of map visualizations: a bubble map that places a bubble over a geographic point, and a shape map that shows the outline of the area that you want to visualize.

Create bubble maps:

To create a bubble map, select the Map option in the Visualization pane. In the Visualizations options, add a value to the Location bucket to use a map visual.

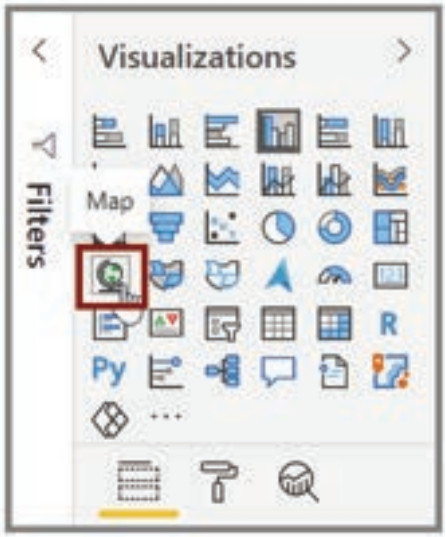


Figure 4.1.10.: Map button on the Visualizations pane.

Power BI accepts many types of location values. It recognizes city names, airport codes, or specific latitude and longitude data. Add a field to the Size bucket to change the size of the bubble for each map location.

Figure 4.1.11.: Resized bubbles on map visual.

Create shape maps:

To create a shape map, select the Filled Map option in the Visualization pane. As with bubble maps, you must add a value to the Location bucket to use this visual. Add a field to the Size bucket to change the intensity of the fill color.

Figure 4.1.12.: Map with different shaded state fills.

A warning icon in the top-left corner of your visual indicates that the map needs more location data to accurately plot values. This is a common problem when the data in your location field is ambiguous, such as using an area name like Washington, which could indicate a state or a district.

One way to resolve the location data problem is to rename your column to be more specific, such as State. Another way is to manually reset the data category by selecting Data Category on the Column tools tab. From the Data Category list, you can assign a category to your data such as "State" or "City."

4.1.5. MATRICES AND TABLES:

You can use Power BI Desktop to create graphical and tabular visuals.

If you have numerical information in a table, such as revenue, a total sum will appear at the bottom. You can manually sort by each column by selecting the column header to switch ascending or descending order. If a column isn't wide enough to display all its contents, select and drag the column header to expand it.

In the Visualizations pane, the order of the fields in the Values bucket determines the order in which they appear in your table.

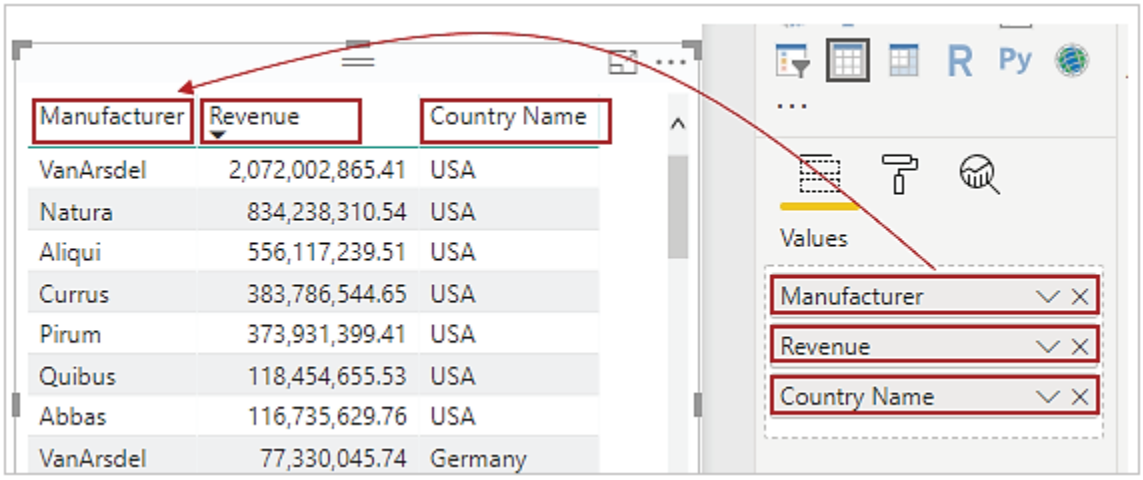


Figure 4.1.1.3.: How values in the Visualization pane match the order of fields on a table.

A matrix is similar to a table, but it has different category headers on the columns and rows. As with tables, numerical information will be automatically totaled along the bottom and right side of the matrix.

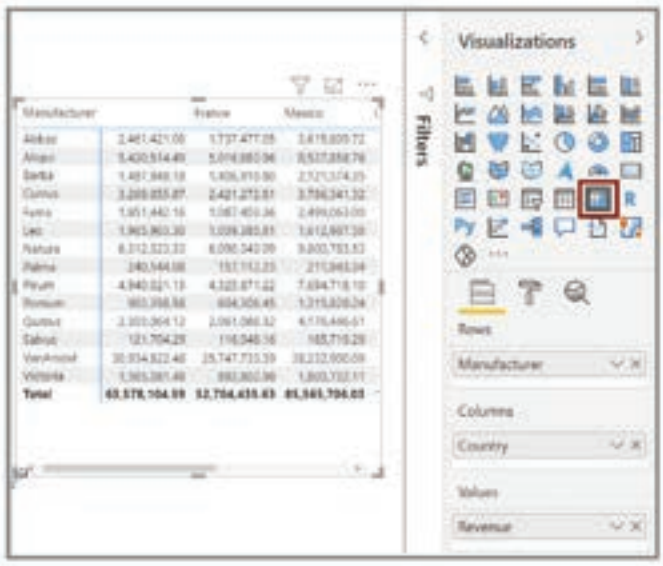


Figure 4.1.14.: Matrix button in the Visualizations pane and the resulting matrix.

Many cosmetic options are available for matrices, such as auto-sizing columns, switching between row and column totals, setting colors, and more.

4.1.6. CREATE SCATTER, WATERFALL, AND FUNNEL CHARTS

Use a scatter chart to compare two different measures, such as unit sales versus revenue.

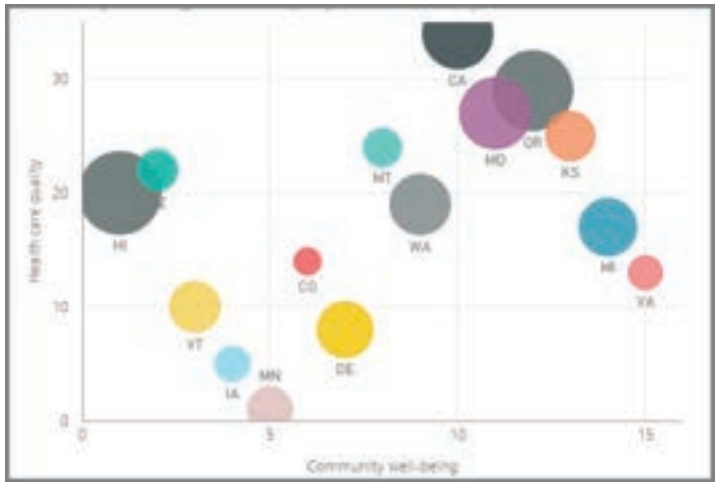


Figure 4.1.15.: A bubble chart

To create a blank chart, select Scatter chart from the Visualizations pane. Drag and drop the two fields that you want to compare from the Fields pane to the X Axis and Y Axis option buckets. At this point, your scatter chart probably has a small bubble in the center of the visual. You need to add a measure to the Details bucket to indicate how you want to segment your data. For example, if you're comparing item sales and revenue, you might want to split the data by category, or manufacturer, or month of sale.

Adding another field to the Legend bucket will color-code your bubbles according to the field's value. You can also add a field to the Size bucket to alter the bubble size according to that value.

Figure 4.1.16.: A color-coded scatter chart.

Scatter charts have many visual formatting options as well, such as turning on an outline for each colored bubble and switching between individual labels. You can change the data colors for other chart types as well.

Figure 4.1.17.: Scatter chart button and the formatting options.

You can create an animation of your bubble chart's changes over time by adding a time-based field to the Play Axis bucket. Select a bubble during an animation to see a trace of its path.

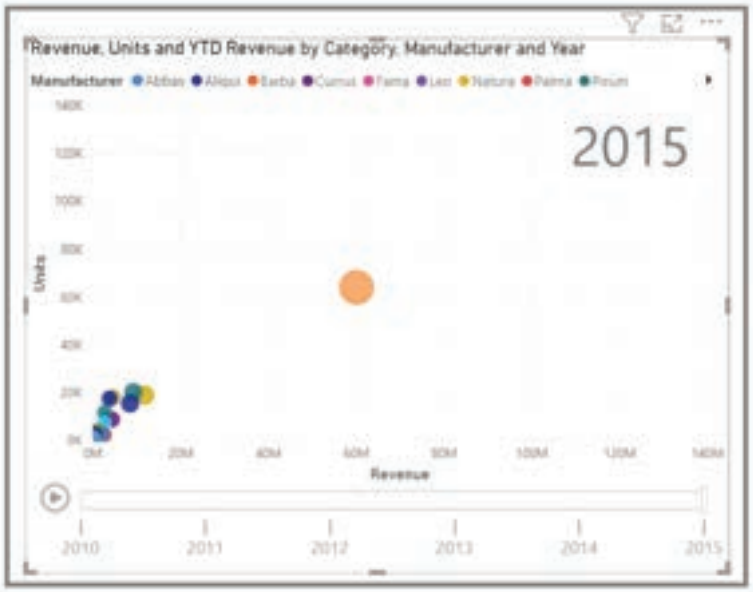


Figure 4.1.18.: An animated bubble chart.

Note:

Remember, if you only see one bubble in your scatter chart, it's because Power BI is aggregating your data, which is the default behavior. To get more bubbles, add a category to the Details bucket in the Visualizations pane.

Create waterfall and funnel charts:

Waterfall and funnel charts are two of the more noteworthy (and uncommon) standard visualizations that are included in Power BI. To create a blank chart of either type, select its icon from the Visualizations pane.

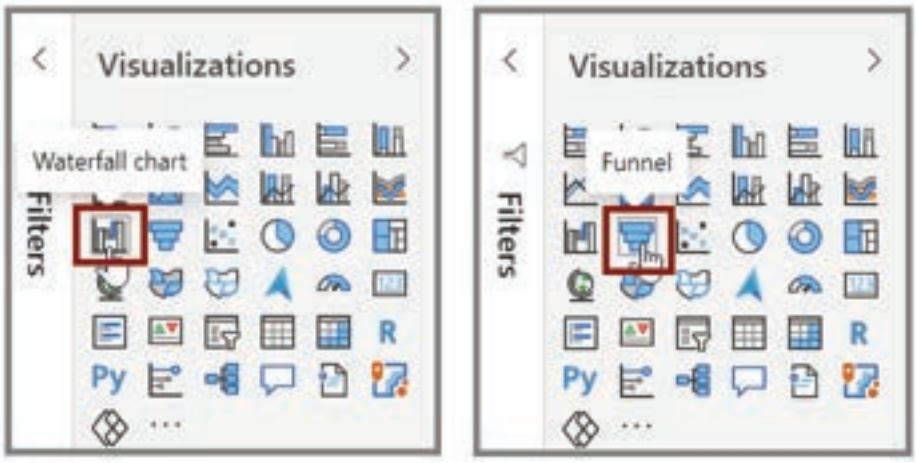


Figure 4.1.19.: Waterfall chart and Funnel buttons on the Visualizations pane.

Waterfall charts are typically used to show changes in a specific value over time.

Figure 4.1.20.: A typical waterfall chart.

Waterfalls only have two bucket options: Category and Y Axis. Drag a time-based field, such as Year, to the Category bucket, and drag the value that you want to track to the Y Axis bucket. Time periods where an increase in value occurred are displayed in green by default, while periods with a decrease in value are displayed in red.

Funnel charts are typically used to show changes over a specific process, such as a sales pipeline or website retention efforts.

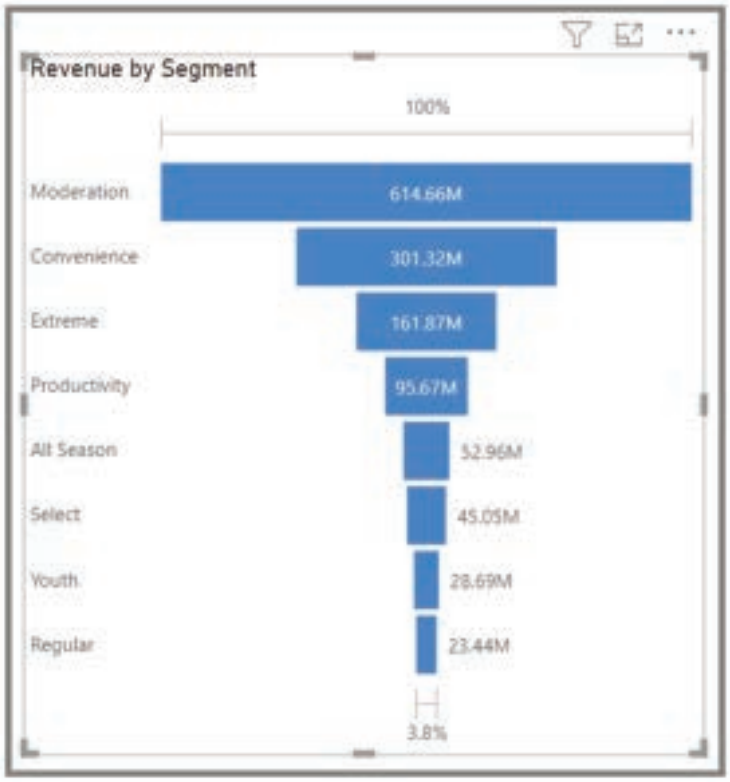


Figure 4.1.21.: A typical funnel chart.

You can slice and customize Waterfall and Funnel charts.

4.1.7. MODIFY COLORS IN CHARTS AND VISUALS

Occasionally, you might want to modify the colors that are used in charts or visuals. Power BI gives you control over how colors are displayed. To begin, select a visual and then select the paintbrush icon in the Visualizations pane.

Figure 4.1.22.: A visual button on the Visualizations pane and its formatting options.

Power BI provides many options for changing the colors or formatting the visual. You can change the color of all bars in a visual by selecting the color picker beside Default color and then selecting your color of choice.

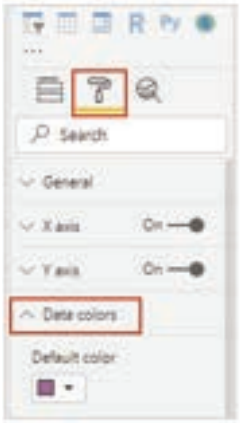


Figure 4.1.23.: Visualization format options for default color.

You can change the color of each bar (or other element, depending on the type of visual that you selected) by turning the Show all slider to On. A color selector will then appear for each element.

Conditional formatting:

You can change the color based on a value or measure. To do so, select the formula box next to Default color.

Figure 4.1.24.: Conditional Formatting option below the Visualizations pane.

The resulting visuals will be colored by the gradient that you select.

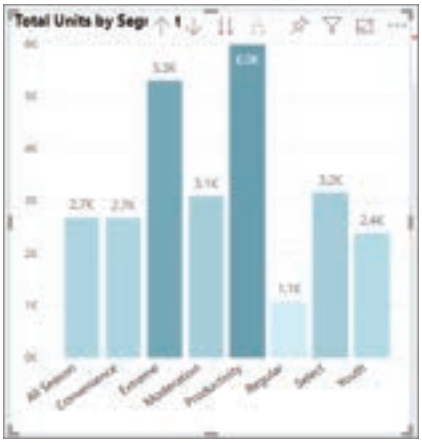


Figure 4.1.25.: A bar chart shaded according to total units by segment.

You can use those values to create rules, for example, to set values above zero to a certain color and values below zero to another color.

In the Analytics pane, you can create many other lines for a visual, such as Min, Max, Average, Median, and Percentile lines.

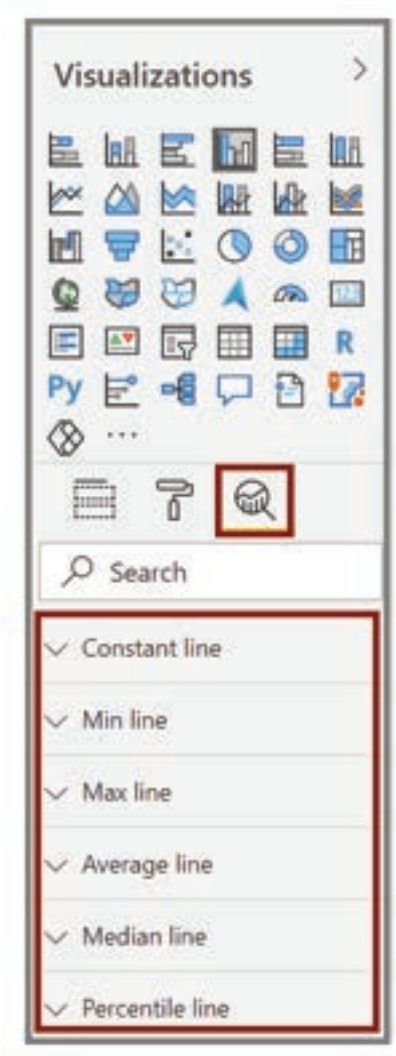


Figure 4.1.26.: Analytics pane below the Visualizations pane.

You can create a border around an individual visualization, and like other controls, you can specify the color of that border as well.

4.1.8. PAGE LAYOUT AND FORMATTING

Power BI Desktop gives you the ability to control the layout and formatting of your report pages, such as size and orientation.

Use the Page View menu from the View tab to change the way that your report pages scale. The available options include Fit To Page (default), Fit To Width, and Actual Size.

Figure 4.1.27.: "Page View" button and the dropdown options.

You can also change the page size. By default, the report page size is set to 16:9. To change the page size, make sure that no visuals are selected, select the paintbrush icon on the Visualizations pane, and then select Page Size to expand that section.

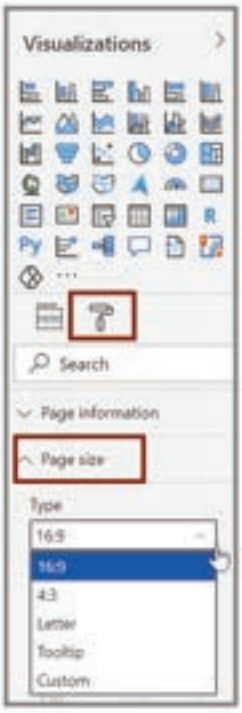


Figure 4.1.28.: "Page size" options under the Visualizations pane.

Options for page size include 4:3 (more of a square aspect ratio) and Dynamic (the page will stretch to fill the available space). A standard letter size option is available for reports as well. You might need to resize your visuals after changing the page size to ensure that they're completely on the canvas.

You can specify a custom page size, setting the size by inches or pixels, and you can also change the background color of the entire report.

Another option is to select Cortana, which sizes the report so that it can be used as a result for searches that use Cortana.

Add static elements:

Along with data-bound visuals, you can also add static elements such as text boxes, images, and shapes to improve the visual design of your reports. To add a visual element, select Text Box, Image, or Shapes from the Home tab.

Figure 4.1.29.: "Text box", "Image", and "Shapes" buttons on the Home tab.

You can display large titles, captions, or short paragraphs in Text boxes, which can also include links and URLs.
Selecting Image will open a file browser where you can select the image from your computer or other networked source. By default, resizing an image in your report will maintain its aspect ratio.
You can insert five types of Shapes, including rectangles and arrows. Shapes can be opaque or transparent with a colored border. The latter is useful for creating borders around groups of visualizations.

Manage how elements overlap:

When you have several elements on a report, Power BI lets you manage how they overlap with each other. This ordering of layers is known as the z-order.

To manage the z-order of elements in a report, select an element and use the Bring forward and Send backward buttons on the Format tab.

Figure 4.1.30.: "Send backward" button and its dropdown options on the Format tab.

Reuse a report layout:

Individual pages of a report can be complex, with multiple visualizations that interact in specific ways and have precise formatting. Occasionally, when building a report, you might want to use the same visuals and layouts for two different pages. For example, if you've just put together a report page on gross revenue, you might want an almost identical page on net revenue.

Recreating all your work would be difficult, but with Power BI Desktop, you can duplicate a report page.

Right-click the tab that you want to copy and then select Duplicate Page.

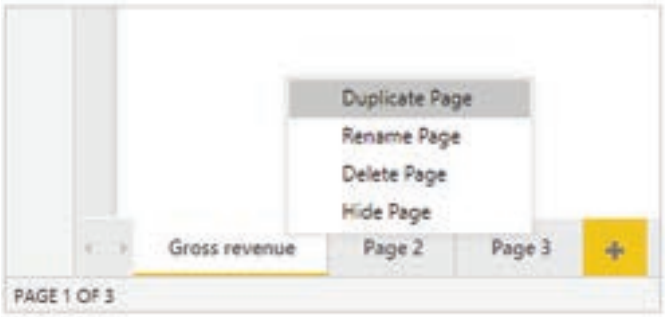


Figure 4.1.31.: "Duplicate Page" option on the bottom of the page.

SUMMARY

- Visuals in Power BI are crucial for presenting data compellingly and insightfully. Power BI offers a wide range of visualizations, including bar charts, pie charts, maps, waterfalls, funnels, and more.
- The module begins with basic visualizations like bar charts and pie charts and progresses to advanced details for report creation.
- Creating visuals in Power BI involves dragging field names onto the canvas or selecting visualization types from the Visualizations pane. You can customize visuals by resizing, moving, or changing their types.
- Slicers are powerful on-canvas filters that segment data by specific values, making it easier to explore information.
- Map visualizations in Power BI include bubble maps and shape maps, which allow for geographic data representation.

- Matrices and tables are used to display numerical data and can be customized to show totals and sort data.
- Scatter, waterfall, and funnel charts are effective for comparing and visualizing data with different measures and scales.
- Modifying colors in charts and visuals is possible through various formatting options, including conditional formatting.
- Page layout and formatting options include adjusting page size, orientation, and adding static elements like text boxes, images, and shapes.
- Managing element overlap and z-order helps organize visuals on report pages effectively.
- You can also duplicate report pages for consistency and efficiency in report creation.

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 What is the primary purpose of using visuals in Power BI?
   a) To create complex reports
   b) To enhance report-creating knowledge
   c) To make reports more colorful
   d) To reduce data complexity

2 Which type of chart is suitable for comparing two or more different values in Power BI?
   a) Scatter chart
   b) Funnel chart
   c) Bubble chart
   d) Waterfall chart

3 How can you add a slicer to your Power BI report?
   a) Drag a field from the Fields pane to the Values bucket.
   b) Select the Slicer button in the Visualizations pane.
   c) Right-click on a visualization and choose "Add Slicer."
   d) Use the Page View menu to add a slicer.

4 What does the Page View menu in Power BI allow you to do?
   a) Change the data source for a report
   b) Change the page layout and formatting
   c) Create new visuals
   d) Add static elements to a report

5 Which type of chart is used to visualize changes in a specific value over time in Power BI?
   a) Funnel chart
   b) Scatter chart
   c) Waterfall chart
   d) Bubble chart

6 What is the purpose of a slicer in Power BI?
   a) To add static images to a report
   b) To filter and segment data in a report
   c) To create complex visuals
   d) To add conditional formatting to a chart

7 How can you change the colors of elements in a visual in Power BI?
   a) By selecting the Default color option in the Visualizations pane
   b) By using the Bring forward and Send backward buttons
   c) By changing the report page size
   d) By dragging and dropping fields onto the canvas

8 What is the purpose of a combination chart in Power BI?
   a) To create complex visuals with multiple layers
   b) To visualize multiple measures with different scales in a single chart
   c) To combine visuals from different data sources
   d) To create a map visualization

9 Which type of visualization allows report users to segment data by a specific value?
   a) Bubble chart
   b) Funnel chart
   c) Slicer
   d) Scatter chart

10 What does the z-order refer to in Power BI?
   a) The order in which visuals are created
   b) The order of layers in a report, determining how elements overlap
   c) The order of fields in the Values bucket
   d) The order of pages in a report

11 How can you duplicate a report page in Power BI?
   a) Use the Duplicate Page button in the Page View menu.
   b) Right-click on a visual and choose Duplicate Page.
   c) Drag and drop the page tab to create a duplicate.
   d) There is no way to duplicate a report page in Power BI.

12 Which type of map visualization in Power BI shows the outline of a geographic area?
   a) Bubble map
   b) Shape map
   c) Heat map
   d) Scatter map

13 What are some examples of cosmetic changes you can make to a visual in Power BI?
   a) Adding slicers and filters
   b) Changing the page layout
   c) Adjusting the data source
   d) Modifying background, alignment, and data colors

14 How can you change the size of a visual in Power BI?
   a) By dragging the handles in or out
   b) By selecting the Default color option
   c) By adding fields to the Visualization pane
   d) By using the Page View menu

15 What is the main purpose of using combination charts in Power BI?
   a) To create complex visuals with multiple layers
   b) To visualize multiple measures with different scales in a single chart
   c) To add conditional formatting to a chart
   d) To create shape maps

Answers

1 *b. To enhance report-creating knowledge*
2 *c. Bubble chart*
3 *b. Select the Slicer button in the Visualizations pane.*
4 *b. Change the page layout and formatting*
5 *c. Waterfall chart*
6 *b. To filter and segment data in a report*
7 *a. By selecting the Default color option in the Visualizations pane*
8 *b. To visualize multiple measures with different scales in a single chart*
9 *c. Slicer*
10 *b. The order of layers in a report, determining how elements overlap*

11 *a. Use the Duplicate Page button in the Page View menu.*
12 *b. Shape map*
13 *d. Modifying background, alignment, and data colors*
14 *a. By dragging the handles in or out*
15 *b. To visualize multiple measures with different scales in a single chart*

SELF-EXAMINATION QUESTIONS FOR PRACTICE:

1 What is the significance of visuals in Power BI, and how do they enhance data presentation?

2 Can you name some common types of visuals available in Power BI for data representation?

3 What is the purpose of the initial part of the module, which covers basic visualizations like bar charts and pie charts?

4 How can you create a new visualization in Power BI Desktop, and what are the two primary methods mentioned in the content?

5 Explain how you can customize visuals in Power BI, including resizing and changing types.

6 What are slicers in Power BI, and how do they contribute to the effectiveness of reports?

7 Describe the different types of map visualizations available in Power BI and their applications.

8 What is the difference between matrices and tables in Power BI, and how can you customize them for data presentation?

9 How can you create scatter, waterfall, and funnel charts in Power BI, and when would you use each of these visualization types?

10 What options are available for modifying colors in charts and visuals in Power BI, and what is conditional formatting used for?

CHAPTER 4

4.2. UNDERSTAND ADVANCED DATA VISUALIZATION CONCEPTS

Often, report creators hear complaints that reports look outdated and take forever to load. You can solve many of these problems with features already available in Power BI. As an enterprise data analyst, implementing these advanced features will make your reports more cohesive, inclusive, and efficient.

LEARNING OBJECTIVES

- Create and import a custom report theme.
- Create custom visuals with R or Python.
- Enable personalized visuals in a report.
- Review report performance using Performance Analyzer.
- Design and configure Power BI reports for accessibility.

4.2.1. CREATE AND IMPORT A CUSTOM REPORT THEME

When you create a custom theme, you ensure a cohesive look and feel for reports across your organization. Deploying an enterprise-wide custom theme provide consistency for:

- Organizational branding
- Accessibility requirements

Access themes:

Navigate to the View tab on the ribbon, then select the drop-down arrow on Themes to:

- Choose a built-in theme
- Browse for themes
- View the Theme gallery
- Customize current theme
- Save current theme

Create your custom theme:

Currently there are almost 20 built-in report themes, including high contrast and color-blind safe options. However, if you have a certain color palette and font family in mind, you can go to Customize current theme. From here, you can make changes in the following categories:

- Theme name and color settings include theme colors, sentiment colors, divergent colors, and structural colors (Advanced).
- Text settings include font family, size, and color, which sets the primary text class defaults for labels, titles, cards and KPIs, and tab headers.

- Visual settings include background, border, header, and tooltips.
- Page element settings include wallpaper and background.
  Filter pane settings include background color, transparency, font and icon color, size, filter cards.

After you make your changes, select Apply to save changes to your theme. Now you can use your theme for your current report and export for future use. Custom themes are an excellent choice for organizations, allowing a cohesive look for reports across entire department or organization.

Using the Customize current theme option is a quick and easy way to create a custom theme. If you want to make finer adjustments to themes you can also create a custom theme through JSON, which we don't cover in this module.

Export and import themes:

After you've applied changes to your custom theme, you need to go back to the Themes drop-down menu and select Save current theme to export the theme. A JSON file will be created that can be shared with others and used for any future reports. Now that your custom theme has been exported, you and others will need to import it to apply to other reports. Importing is as easy as exporting is – navigate to the Themes drop-down menu again, select Browse for themes, and choose the JSON file you just created (or that was shared with you). You'll get a notification when the theme successfully imports.

Enterprise considerations:

Any changes made to this theme will need to be saved again to either overwrite the existing theme or as a new theme. Themes are local to the file as well, so changes made on someone's copy on their computer won't affect your copy. The theme could be saved as a Power BI Template (.PBIT) file, and then shared that way as well. When sharing custom themes, consider a business process to validate that the theme and/or template are being used.

4.2.2. ENABLE PERSONALIZED VISUALS IN A REPORT

In a larger organization, it can be difficult to design a single report that meets everyone's needs, so one solution is using personalized visuals. These allow individual consumers to make minor design changes to best understand the data themselves.

Tip:

This is a powerful feature for more advanced report consumers. Empower your users by sharing the Power BI Consumer Documentation for personalized visuals.

What can consumers change:

Report consumers can make changes for the following items:

- Visualization type
- Swap measure or dimension
- Add/remove legend
- Compare multiple measures
- Change aggregations

Consumers can create a personal bookmark to save their changes after personalizing a report, and even share changes with others. Best of all, they can always reset to default to restore the report view.

Enable personalized visuals:

You can enable personalized visuals in Power BI Desktop or Power BI service.

- In Power BI Desktop, go to File > Options and settings > Options > Current file > Report settings. Make sure Personalize visuals is turned on.
- In Power BI service, go to Settings on the specific report, then toggle Personalize Visuals on, and Save.

By default, when you enable personalized visuals, it's enabled for all visuals in a report. You can modify each visual to allow personalization or not. You'll need to enable this setting for each report.

Tip:

Use a Power BI Template (.PBIT) file to enable personalized visuals and add on top of your custom theme for faster report building in the future.

4.2.3. DESIGN AND CONFIGURE POWER BI REPORTS FOR ACCESSIBILITY

Accessibility is at the heart of Microsoft's values, and Power BI is committed to making Power BI experiences as accessible to as many people as possible. As an enterprise report creator, it's up to you to use the available tools to create inclusive reporting across the organization.

Note:

This unit highlights some accessible design features, and more details including a checklist, can be found in the Power BI Documentation.

Consider your audience:

Easily improve accessibility for most consumers with the following modifications:

- Alt text for visuals, shapes, and images
- Set tab order for visuals, shapes, and images
- Consistent font, colors, positioning
- Colorblind-friendly color schemes
- Using text or icons in addition to color
- Avoid jargon and acronyms
- Set sort order for visuals
- Disable auto-start videos and audio
- Provide captions and transcripts for videos and audio
- Avoid excess decorative shapes and images

Alt text:

Alt text is one of the most helpful and easily configurable accessibility features. For all visuals, shapes, and images, you can add alt text - a description for use with screen readers.

Screen readers will automatically read the title and type of visual, so add insight and context for low vision users, such as "Total sales by category, further broken down by product". If you want to include specific data points, instead of adding them to the static alt text, you can use DAX measures and conditional formatting to create dynamic alt text in enrich the user experience.

To set alt text, first select the visual or image, navigate to the Format visual/shape menu, and select Alt text. To add dynamic alt text, select the Fx button.

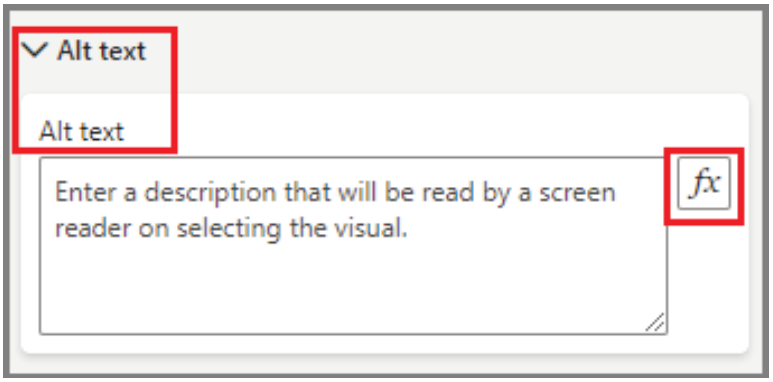


Figure 4.2.1.: Alt text menu with three ordered items and one hidden item.

Set tab order:

In Power BI Desktop, you can set the tab order for how a keyboard user will experience your report. First, navigate to the View tab in the ribbon, then select Selection in the Show Panes section.

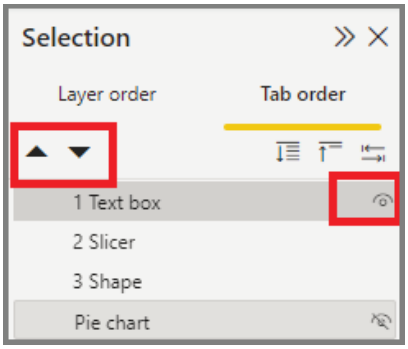


Figure 4.2.2.: Tab order menu with three ordered items and one hidden item.

You'll see Layer order and Tab order, with a list of report elements. Layer order allows you to stack elements, whereas tab order dictates which item will be accessed next when the keyboard user tabs to the next item. Use the up/down arrows to set the order. Hide items by clicking the eye icon. This action will move the item to the bottom, remove the numbered position, and put a line through the eye to indicate hidden state.

Tip:

Set tab order and turn off tab order (mark the item as hid`den) on any decorative items.

4.2.4. CREATE CUSTOM VISUALS WITH R OR PYTHON

If your organization relies on open-source languages like R and Python, you can write scripts to transform data and create visualizations. Using R and Python visuals in Power BI extends the opportunity to collaborate with Data Scientists in your organization, and enables the use of visuals your data team may already be using.

Note:

This module won't go in-depth for creating R and Python visuals, however you can review the Add an R or Python visual Microsoft Learn unit for more detail.

R visuals:

In order to develop R visuals, you'll need a supported version of R installed first. Then add the R visual to the canvas, just like using a table or bar chart visual, and add your R code within the visual.

Python visuals:

To develop Python visuals, you need to have Python installed on your machine. Many Python packages are supported in Power BI, but not all.

4.2.5. REVIEW REPORT PERFORMANCE USING PERFORMANCE ANALYZER

A good report designer is always looking for ways to improve the end-user experience, and report speed is a common complaint. In an enterprise with many reports, every millisecond can improve the experience. By using Performance Analyzer, you can identify suboptimal report elements and adjust where possible.

What is Performance Analyzer?

Performance Analyzer is a built-in feature in Power BI Desktop that measures how long report elements take to update and refresh, allowing you to see if certain elements are significantly slower.

Monitor with Performance Analyzer:

In Power BI Desktop, navigate to the View ribbon, then select Performance Analyzer. The Performance Analyzer pane will appear to the right of the canvas. When you're ready to interact with the report, select the Start Recording button, perform report actions like adjusting slicers, highlighting values, changing pages, etc. You can also Refresh visuals, and then Stop when you're done.

Some easy ways to improve performance include:

- Limit visuals per page. If the entire page is slow to load, consider spreading visuals across multiple pages instead.
- Remove unnecessary columns and rows. For slower queries, review the data and determine if anything can be removed.
- Set data types. By default, Power BI will assume data types for imported columns, which should be verified, and all new columns must be manually set.
- Apply most restrictive filters. More data will take more time to load, and isn't always the best user experience either. Consider limited results first, and allow drilling for more details.

Tip:

The best time to optimize your report is before sharing it with end-users as it allows you to improve performance before soliciting user feedback. However, improvement is a continual process and can and should be done regularly once the report is live as well.

4.2.6. DATA EXTRACTION IN POWER BI

Data extraction in Power BI is a fundamental step in the process of building reports and dashboards. To extract data in Power BI, you typically connect to one or more data sources, import or load data into Power BI, and then shape and transform the data as needed. Here's a step-by-step guide to data extraction in Power BI:

Launch Power BI Desktop:

Start by opening the Power BI Desktop application on your computer.

Get Data:

Click on the "Get Data" option in the “Home” tab on the Power BI Desktop ribbon. This will open a dialog box where you can choose your data source.

Select Data Source:

Choose the type of data source you want to connect to. Power BI supports various data sources, including databases, files, online services, and more. Some common data source options include:

- Database: You can connect to SQL Server, Azure SQL Database, MySQL, Oracle, and other databases.
- Files: You can import data from Excel, CSV, text files, or other file formats.
- Online Services: Connect to cloud services like SharePoint, Dynamics 365, Salesforce, and web services.
- Online Content: Extract data from websites or online sources using web connectors.

Connect to Data:

After selecting the data source, you'll need to provide connection details. This may include server names, database credentials, file paths, URLs, or authentication methods, depending on the data source.

Load or Transform Data:

Once connected, you can choose to load the data directly into Power BI or apply transformations before loading. Common data transformations include filtering rows, removing columns, merging tables, and creating calculated columns.
To transform data, click on the "Edit" button. This opens the Power Query Editor, where you can perform data shaping and transformation tasks.

Data Modeling:

After loading or transforming the data, you can model the data by defining relationships between tables, creating measures, and organizing fields into hierarchies. This step is crucial for building meaningful reports.

Data Refresh:

Power BI allows you to schedule data refresh to ensure that your reports stay up-to-date. Depending on your data source, you can set up a data refresh frequency to automatically update your reports with the latest data.

Save Your Power BI File:

Save your Power BI file (.PBIX) to retain your data source connections, data transformations, and report designs. This file can be reopened and refreshed at any time.

Create Visualizations:

With your data loaded into Power BI, you can start creating visualizations, reports, and dashboards using the data you've extracted and modeled.

Publish to Power BI Service:

If you want to share your reports with others, you can publish them to the Power BI Service. This allows you to collaborate, share, and access your reports from the cloud.

Data Security:

Be mindful of data security and access control. Power BI offers features to secure your data, such as row-level security and data encryption.

SUMMARY

- Custom report themes in Power BI provide a consistent look and feel for reports across an organization, benefiting organizational branding and accessibility requirements.
- To access themes, users can navigate to the View tab and select Themes to choose built-in themes, browse for themes, view the Theme gallery, customize the current theme, or save the current theme.
- Custom themes allow extensive modifications, including theme colors, text settings, visual settings, page element settings, and filter pane settings.
- After customizing a theme, users can apply it and export it as a JSON file for future use. Custom themes are useful for maintaining a cohesive look across an organization's reports.
- Data extraction is a fundamental step in creating Power BI reports and dashboards.
- The process involves launching Power BI Desktop, selecting a data source, connecting to it, and loading or transforming the data.
- Power BI supports various data sources, including databases, files, online services, and web content.
- After connecting to the data source, users can apply transformations to clean and shape the data using the Power Query Editor.
- Data modeling includes defining relationships between tables and creating measures for meaningful reports.
- Users can schedule data refresh to keep reports up-to-date and save Power BI files (.PBIX) to retain data source connections and transformations.
- Visualizations and reports can be created with the loaded and transformed data, and reports can be published to the Power BI Service for collaboration.
- Data security and access control considerations are essential throughout the data extraction and reporting process.

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 What is the primary purpose of creating a custom theme in Power BI?
   a) Enhancing report performance
   b) Ensuring data accuracy
   c) Ensuring a cohesive look and feel for reports
   d) Adding custom visuals to reports

2 Which of the following is NOT a category that you can customize when creating a custom theme in Power BI?
   a) Theme name and color settings
   b) Text settings
   c) Visual settings
   d) Data extraction settings

3 How can you access the themes in Power BI?
   a) From the Home tab
   b) From the View tab
   c) From the Data tab
   d) From the Insert tab

4 Which file format is used to export and share custom themes in Power BI?
   a) PDF
   b) PNG
   c) JSON
   d) CSV

5 What does the "Personalize Visuals" feature in Power BI allow individual consumers to do?
   a) Change the data source of a report
   b) Customize the report layout
   c) Make minor design changes to better understand the data
   d) Share reports with others

6 Which of the following can consumers change when using the "Personalize Visuals" feature in Power BI?
   a) Report authorship
   b) Visualization type
   c) Report title
   d) Data source connection

7 How can you enable personalized visuals in Power BI Desktop?
   a) From the File menu, select "Export"
   b) From the View tab, select "Performance Analyzer"
   c) From the Options menu, select "Personalize visuals"
   d) From the Data tab, select "Import Data"

8 What is one way to improve report performance in Power BI?
   a) Add more visuals to a page
   b) Increase the data size
   c) Limit visuals per page
   d) Use complex DAX calculations

9 Which tool in Power BI Desktop measures how long report elements take to update and refresh?
   a) Data Extractor
   b) Performance Analyzer
   c) Report Designer
   d) Theme Builder

10 What is Alt text used for in Power BI?
   a) Adding alternative text to images and visuals
   b) Creating dynamic alt text for data points
   c) Defining report layout
   d) Customizing report colors

11 In Power BI Desktop, how can you set the tab order for keyboard navigation?
   a) From the Format menu
   b) From the Home tab
   c) From the View tab
   d) From the Data tab

12 What is the purpose of setting tab order for visuals in Power BI?
   a) To control the printing order of visuals
   b) To determine which visual is displayed first in a report
   c) To specify the order in which keyboard users navigate through visuals
   d) To apply color formatting to visuals

13 Which scripting languages can be used to create custom visuals in Power BI?
   a) JavaScript and HTML
   b) Python and R
   c) SQL and C#
   d) Ruby and Perl

14 What is the purpose of data refresh in Power BI?
   a) To extract data from external sources
   b) To transform data into visualizations
   c) To schedule automatic updates of report data
   d) To create custom visuals

15 Which of the following is NOT a best practice for designing Power BI reports for accessibility?
   a) Providing captions and transcripts for videos and audio
   b) Using colorblind-friendly color schemes
   c) Including jargon and acronyms
   d) Setting sort order for visuals

Answers

1 *c) Ensuring a cohesive look and feel for reports*
2 *d) Data extraction settings*
3 *b) From the View tab*
4 *c) JSON*
5 *c) Make minor design changes to better understand the data*
6 *b) Visualization type*
7 *c) From the Options menu, select "Personalize visuals"*
8 *c) Limit visuals per page*
9 *b) Performance Analyzer*
10 *a) Adding alternative text to images and visuals*
11 *c) From the View tab*
12 *c) To specify the order in which keyboard users navigate through visuals*
13 *b) Python and R*
14 *c) To schedule automatic updates of report data*
15 *c) Including jargon and acronyms*

SELF-EXAMINATION QUESTIONS FOR PRACTICE:

1 How can creating custom report themes benefit organizations in Power BI reports?

2 What are the steps involved in customizing a report theme in Power BI, and what elements can you customize?

3 What is the purpose of enabling personalized visuals in Power BI, and what options do report consumers have for customization?

4 How can you make Power BI reports more accessible, and what considerations should you keep in mind when designing for accessibility?

5 Briefly explain how to add alt text to visuals, shapes, and images in Power BI for accessibility.

6 In Power BI Desktop, where can you find the Performance Analyzer tool, and what is its primary purpose?

7 What are some performance optimization tips mentioned in the content for improving the speed of Power BI reports?

CHAPTER 5
CREATING REPORTS AND OUTPUT OPTIONS

5.1. CREATE AND SHARE YOUR FIRST POWER BI REPORT

With Power BI you can create compelling visuals and reports. In this module you learn how to use Power BI Desktop to connect to data, build visuals, and create a report that you can share with others in your organization. Then you learn how to publish the report to the Power BI service, and let others see your insights and benefit from your work.

LEARNING OBJECTIVES

- Create a report in Power BI
- Share Power BI reports

5.1.1. INTRODUCTION

Welcome to the learning module designed to get you up and running with Microsoft Power BI Desktop. In this module, you'll learn how to get around in Power BI Desktop, connect to data, create visuals and reports, and publish those reports to the Power BI service.

Power BI Desktop lets you create a collection of queries, data connections, and reports that can easily be shared with others. Power BI Desktop integrates proven Microsoft technologies—the powerful Microsoft Power Query for Excel engine, data modeling, and visualizations—and works seamlessly with the online Power BI service.

Through the combination of Power BI Desktop (where analysts and others can create powerful data connections, models, and reports) and the Power BI service (where Power BI Desktop reports can be shared, so that users can view and interact with them), new insights from the world of data are easier to model, build, share, and extend.

Data analysts will find Power BI Desktop a powerful, flexible, and a highly accessible tool for connecting with and shaping the world of data, building robust models, and crafting well-structured reports.

How Power BI Desktop works:

With Power BI Desktop, you connect to data (usually multiple data sources), shape that data (through queries that build insightful, compelling data models), and use that model to create reports (which others can use, build upon, and share).

When these steps—connect, shape, and report—are finished to your satisfaction, you can save your work in the Power BI Desktop file format, which uses the .pbix extension. Power BI Desktop files can be shared like any other file, but the most compelling way to share them is to upload them to the Power BI service.

Power BI Desktop centralizes, simplifies, and streamlines what can otherwise be a scattered, disconnected, and arduous process of designing and creating business intelligence repositories and reports.

Ready to give it a try? Then let's get started.

Install and run Power BI Desktop:

You can download Power BI Desktop from the Power BI service by selecting the three dots, then download and then selecting Power BI Desktop.

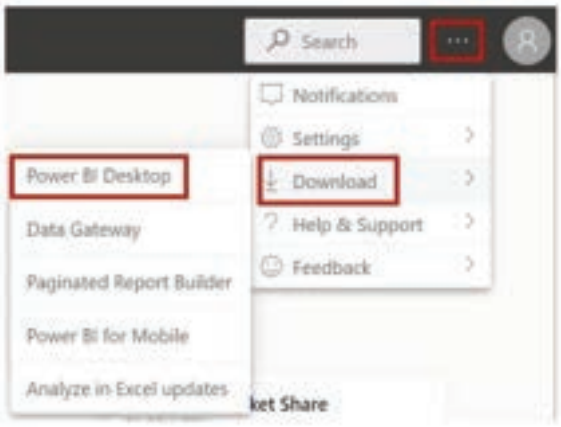


Figure 5.1.1.: Install Power BI Desktop from the Power BI service

Power BI Desktop is installed as an application and runs on your desktop. When you start Power BI Desktop, a Welcome screen is shown.

Figure 5.1.2.: Install Power BI Desktop from Microsoft Store

Figure 5.1.3.: Power BI Desktop runs as an application

You can get data, see recent sources, or open other reports directly from the Welcome screen, by using the links in the left pane. If you close the Welcome screen (by selecting the X in the upper-right corner), the Report view of Power BI Desktop is shown.

Figure 5.1.4.: Power BI Desktop empty canvas

With Power BI Desktop you can connect to the world of data, create compelling and foundational reports, and share your efforts with others – who can then build on your work, and expand their business intelligence efforts.

Power BI Desktop has three views:

- **Report view –** You can use queries that you create to build compelling visualizations, arranged as you want them to appear, and with multiple pages, that you can share with others.
- **Data view –** See the data in your report in data model format, where you can add measures, create new columns, and manage relationships.
- **Model view –** Get a graphical representation of the relationships that are established in your data model, and manage or modify them as needed.

Access these views by selecting one of the three icons along the left side of Power BI Desktop. In the following image, Report view is selected, indicated by the yellow band beside the icon.

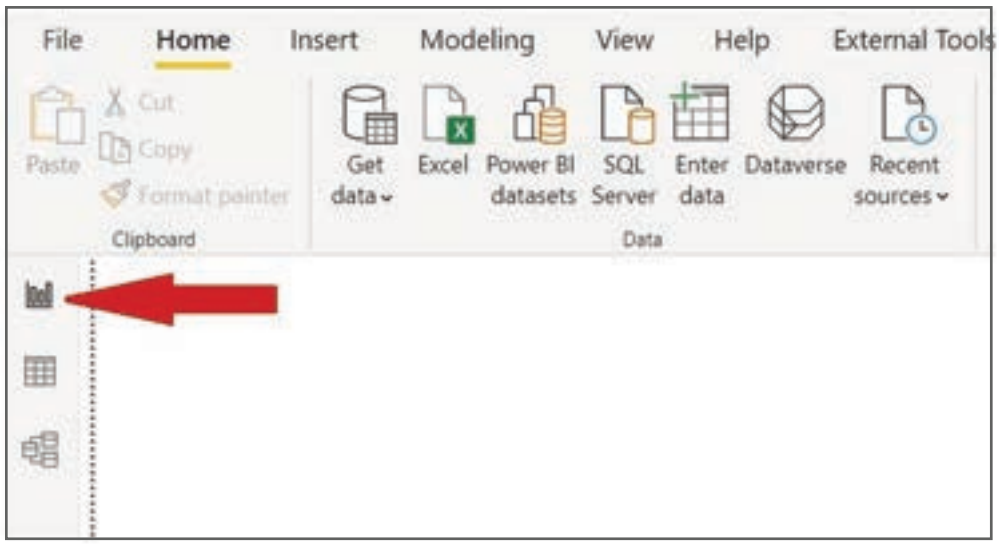


Figure 5.1.5.: Three different views in Power BI Desktop

Power BI Desktop also includes Power Query Editor, which starts in a separate window. In Power Query Editor, you can build queries and transform data, load that refined data model into Power BI Desktop, and create reports.

Now that Power BI Desktop is installed, you're ready to connect to data, shape data, and build reports (usually in that order). In the following units, we'll take a tour through each of those activities in turn.

Power Query Editor:

To get to Power Query Editor, select Transform data from the Home tab of Power BI Desktop.

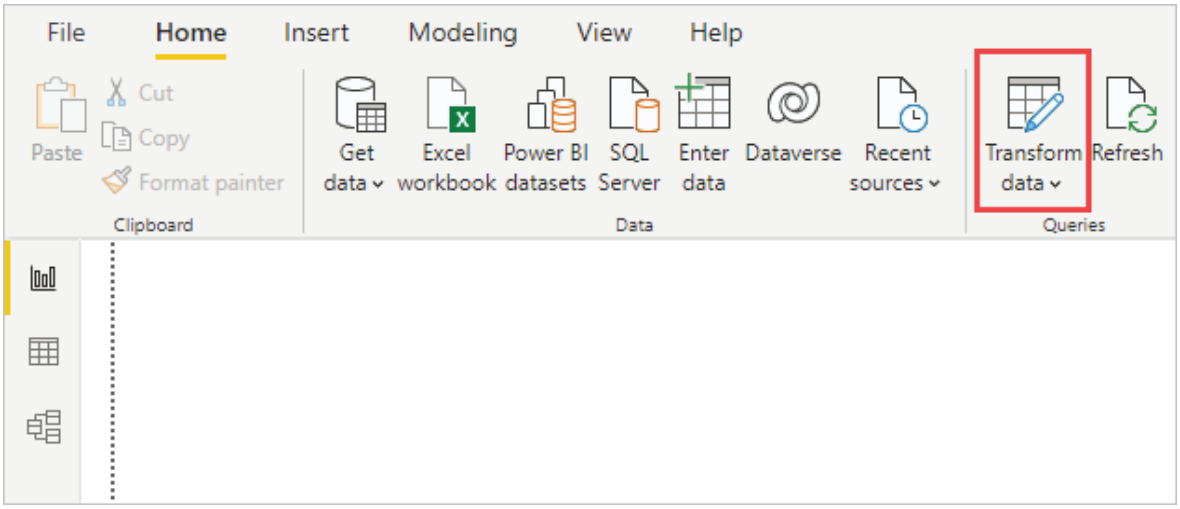


Figure 5.1.6.: Power BI Desktop with the transform data icon highlighted.

With no data connections, Power Query Editor appears as a blank pane, ready for data.

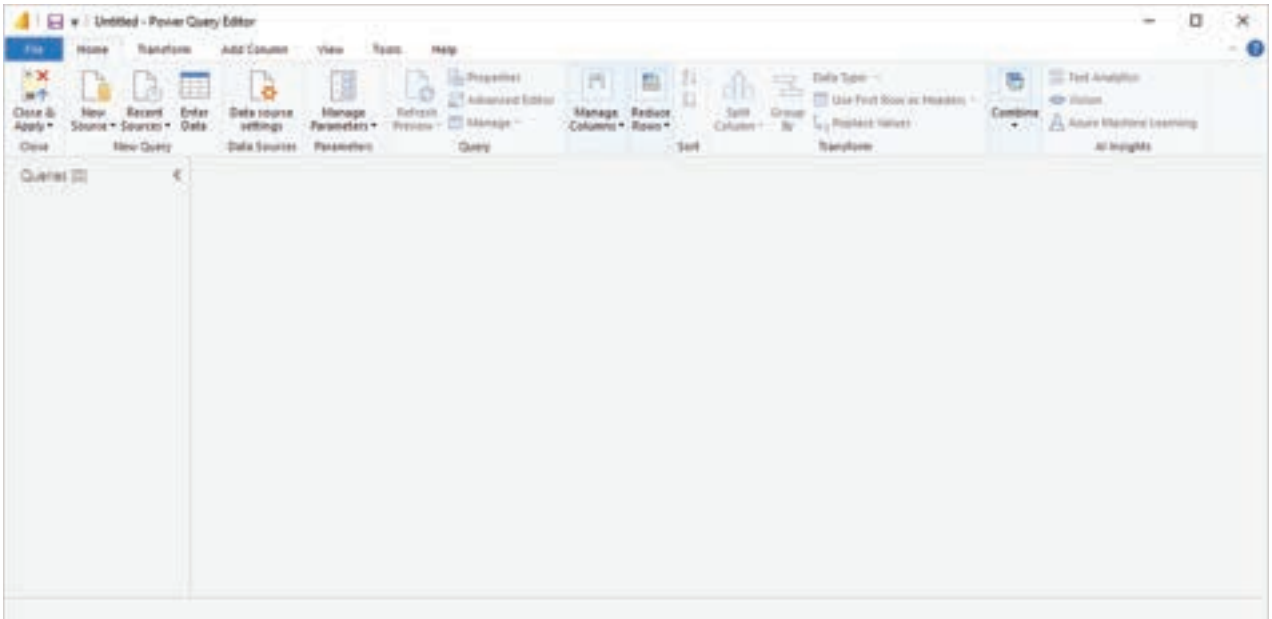


Figure 5.1.7.: Power BI Desktop showing Power Query Editor with no data connections.

After a query is loaded, Power Query Editor view becomes more interesting. If you connect to the following Web data source using the New Source button in the top left, Power Query Editor loads information about the data, which you can then begin to shape:

https://www.bankrate.com/retirement/best-and-worst-states-for-retirement/

Here's how Power Query Editor appears after a data connection is established:

- In the ribbon, many buttons are now active to interact with the data in the query.
- In the left pane, queries are listed and available for selection, viewing, and shaping.
- In the center pane, data from the selected query is displayed and available for shaping.
- The Query Settings pane appears, listing the query's properties and applied steps.

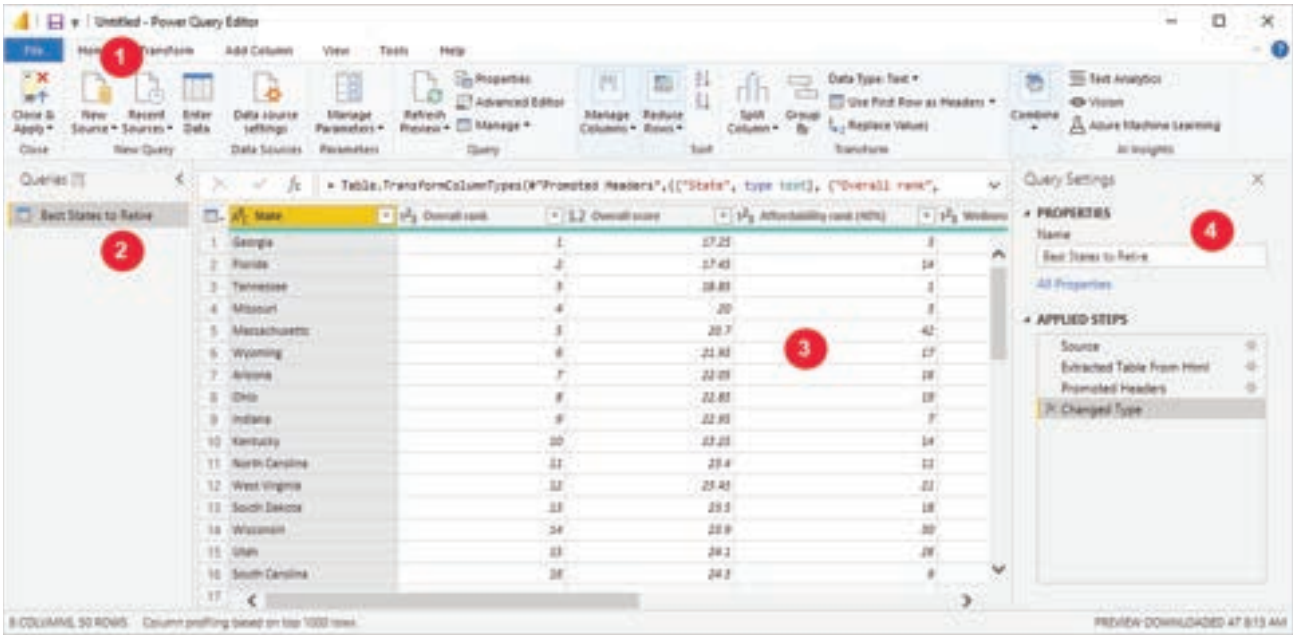


Figure 5.1.8.: Power BI Desktop showing the Query Settings pane in Power Query Editor.

Each of these four areas will be explained later: the ribbon, the Queries pane, the Data view, and the Query Settings pane.

The query ribbon:

The ribbon in Power Query Editor consists of four tabs: Home, Transform, Add Column, View, Tools, and Help.

The Home tab contains the common query tasks.

Figure 5.1.9.: Power BI Desktop showing the Power Query Editor query ribbon.

To connect to data and begin the query building process, select New Source. A menu appears, providing the most common data sources.

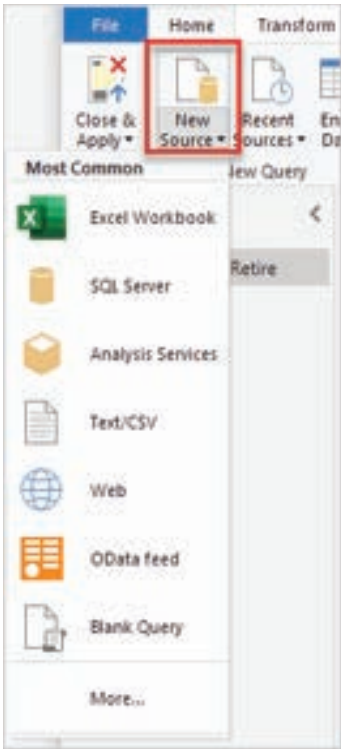


Figure 5.1.10.: Power BI Desktop showing the New Source button.

The Transform tab provides access to common data transformation tasks, such as:

- Adding or removing columns
- Changing data types
- Splitting columns
- Other data-driven tasks

Figure 5.1.11.: Power BI Desktop showing the Transform tab.

The Add Column tab provides more tasks associated with adding a column, formatting column data, and adding custom columns. The following image shows the Add Column tab.

Figure 5.1.12.: Power BI Desktop showing the Add Column tab.

The View tab on the ribbon is used to toggle whether certain panes or windows are displayed. It's also used to display the Advanced Editor. The following image shows the View tab.

Figure 5.1.13.: Power BI Desktop showing the View tab.

It's useful to know that many of the tasks available from the ribbon are also available by right-clicking a column, or other data, in the center pane.

The left (Queries) pane:

The left pane, or Queries pane, displays the number of active queries and the name of the query. When you select a query from the left pane, its data is displayed in the center pane, where you can shape and transform the data to meet your needs. The following image shows the left pane with a query.

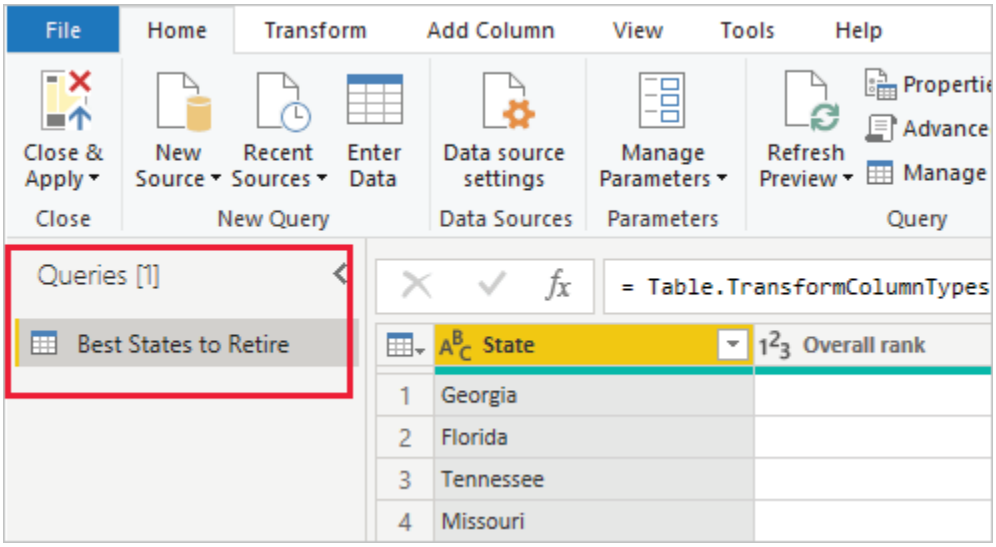


Figure 5.1.14.: Power BI Desktop showing Queries in the left pane.

The center (Data) pane:

In the center pane, or Data pane, data from the selected query is displayed. This pane is where much of the work of the Query view is accomplished.

The following image shows the Web data connection established earlier. The Overall score column is selected, and its header is right-clicked to show the available menu items. Notice that many of these items in the right-click menu are the same as buttons in the ribbon tabs.

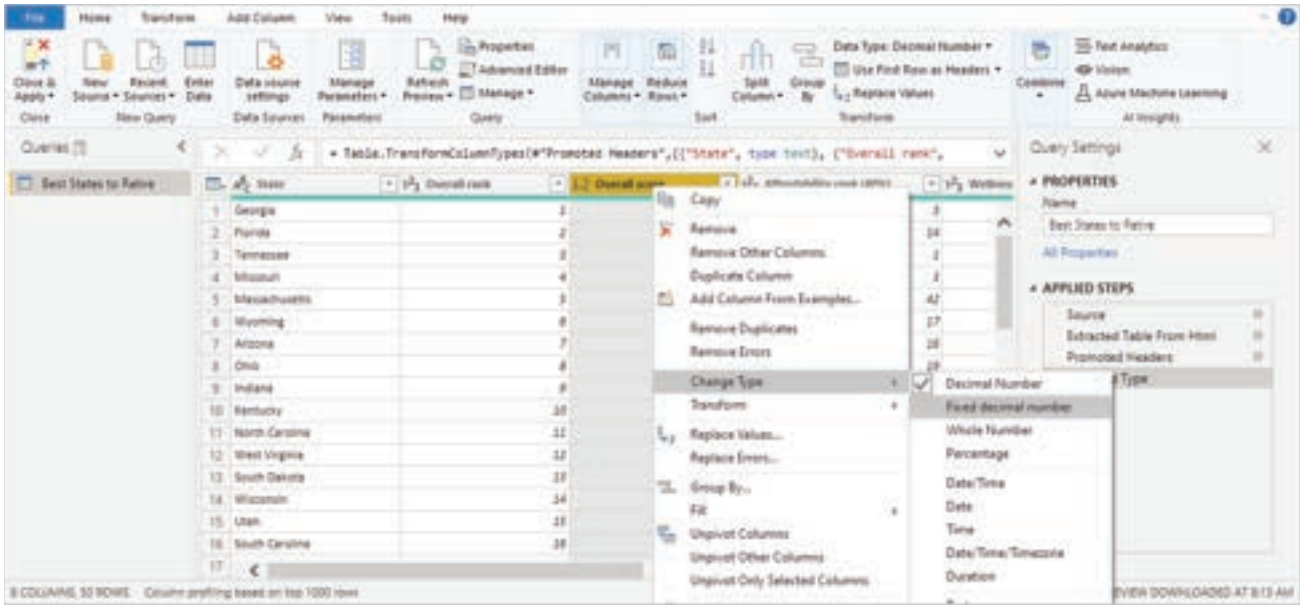


Figure 5.1.15.: Power BI Desktop showing Data in the center pane.

When you select a right-click menu item (or a ribbon button), the query applies the step to the data. It also saves step as part of the query itself. The steps are recorded in the Query Settings pane in sequential order, as described in the next section.

The right (Query Settings) pane:

The right pane, or Query Settings pane, is where all steps associated with a query are displayed. For example, in the following image, the Applied Steps section of the Query Settings pane reflects the fact that we just changed the type of the Overall score column.

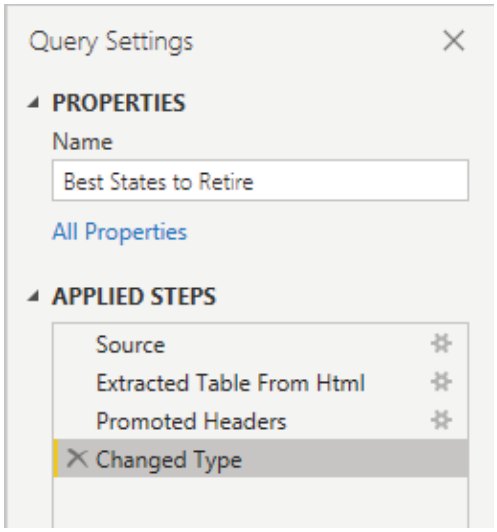


Figure 5.1.16.: Power BI Desktop showing Query Settings in the right pane.

As more shaping steps are applied to the query, they're captured in the Applied Steps section.

It's important to know that the underlying data isn't changed. Rather, Power Query Editor adjusts and shapes its view of the data. It also shapes and adjusts the view of any interaction with the underlying data that occurs based on Power Query Editor's shaped and modified view of that data.

In the Query Settings pane, you can rename steps, delete steps, or reorder the steps as you see fit. To do so, right-click the step in the Applied Steps section, and choose from the menu that appears. All query steps are carried out in the order they appear in the Applied Steps pane.

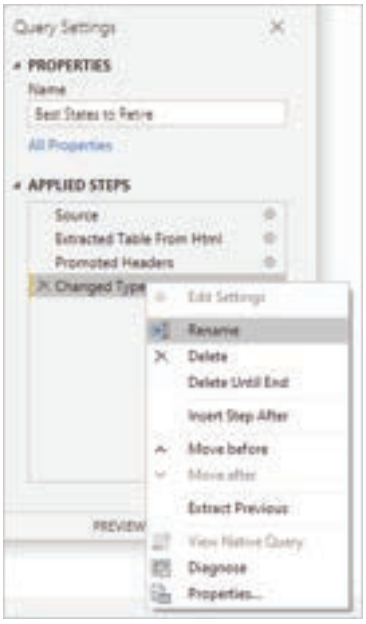


Figure 5.1.17.: Power BI Desktop showing Query Settings Properties and Applied Steps filters.

Advanced Editor:

The Advanced Editor lets you see the code that Power Query Editor is creating with each step. It also lets you create your own code in the Power Query M formula language. To launch the advanced editor, select View from the ribbon, then select Advanced Editor. A window appears, showing the code generated for the selected query.

Figure 5.1.18.: Power BI Desktop showing the Advanced Editor dialog box.

You can directly edit the code in the Advanced Editor window. To close the window, select the Done or Cancel button.

Saving your work:

When your query is where you want it, select Close & Apply from Power Query Editor's File menu. This action applies the changes and closes the editor.

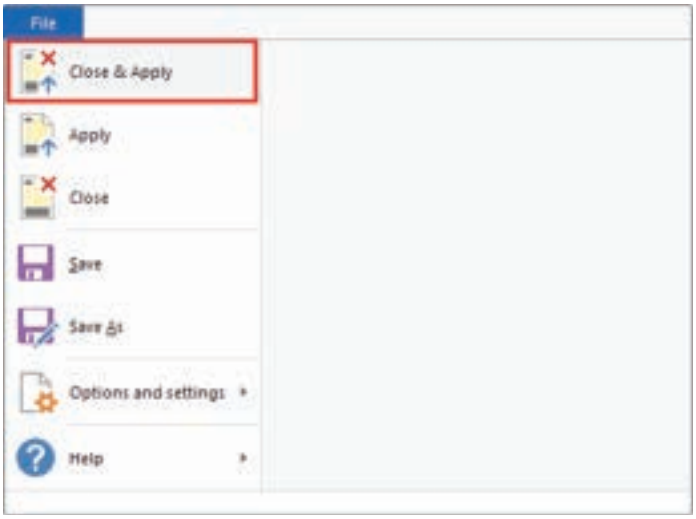


Figure 5.1.19.: Power BI Desktop showing the Close and Apply option under the File tab.

As progress is made, Power BI Desktop provides a dialog to display its status.

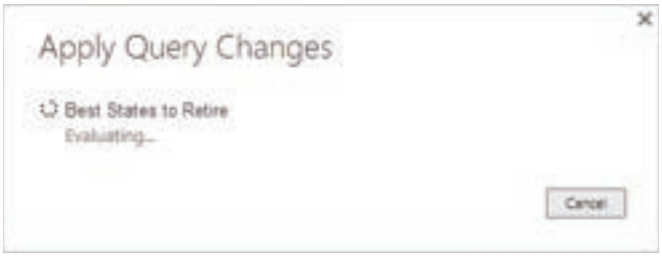


Figure 5.1.20.: Power BI Desktop showing the Applied Query Changes confirmation dialog box.

When you're ready, Power BI Desktop can save your work in the form of a .pbix file.

To save your work, select File > Save (or File > Save As), as shown in the following image.

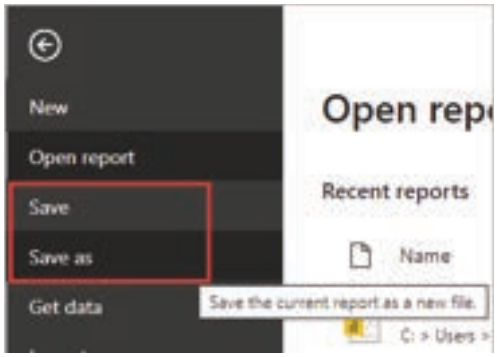


Figure 5.1.21.: Power BI Desktop showing the Power Query Editor File tab. The save and save as options highlighted.

5.1.2. CONNECT TO DATA SOURCES IN POWER BI DESKTOP

Now that Microsoft Power BI Desktop is installed, you're ready to connect to the ever-expanding world of data. There are all sorts of data sources available in the Microsoft Power Query for Excel window. The following image shows how to connect to data by selecting the Home tab on the ribbon and then selecting Get Data > More.

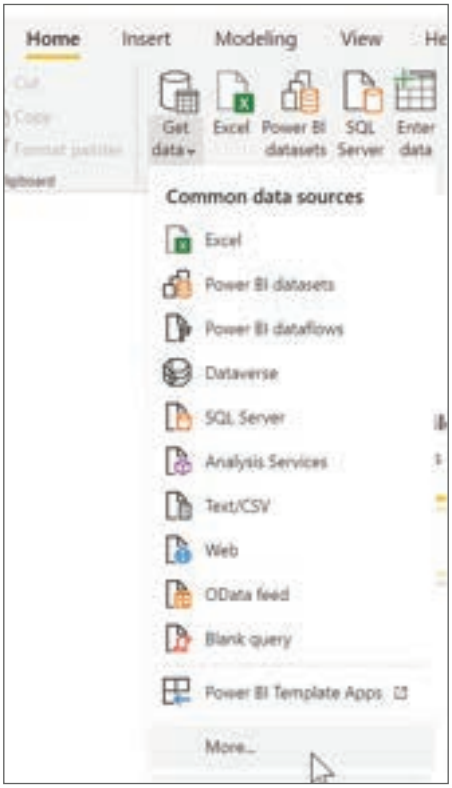


Figure 5.1.22.: Get data

In this unit, we'll connect to a couple different Web data sources.

Imagine you're retiring – you want to live where there's lots of sunshine, low crime rates, and good health care – or perhaps you're a data analyst, and you want that information to help your customers. For example, maybe you want to help your sunglasses retailer target sales where the sun shines most frequently.

Either way, the following web resource has interesting data about those topics, and more:

Best places to retire state rank

Select Get Data > Web, and paste the address:
https://www.bankrate.com/retirement/best-and-worst-states-for-retirement/

Figure 5.1.23.: Connect to web data

When you select OK, the Query functionality of Power BI Desktop goes to work. Query contacts the web resource, and the Navigator window shows what it found on that webpage. In this case, it finds a table (Ranking of best and worst states for retirement). We're interested in the ** Best States to Retire** table, so select it in the list. The Navigator window shows a preview.

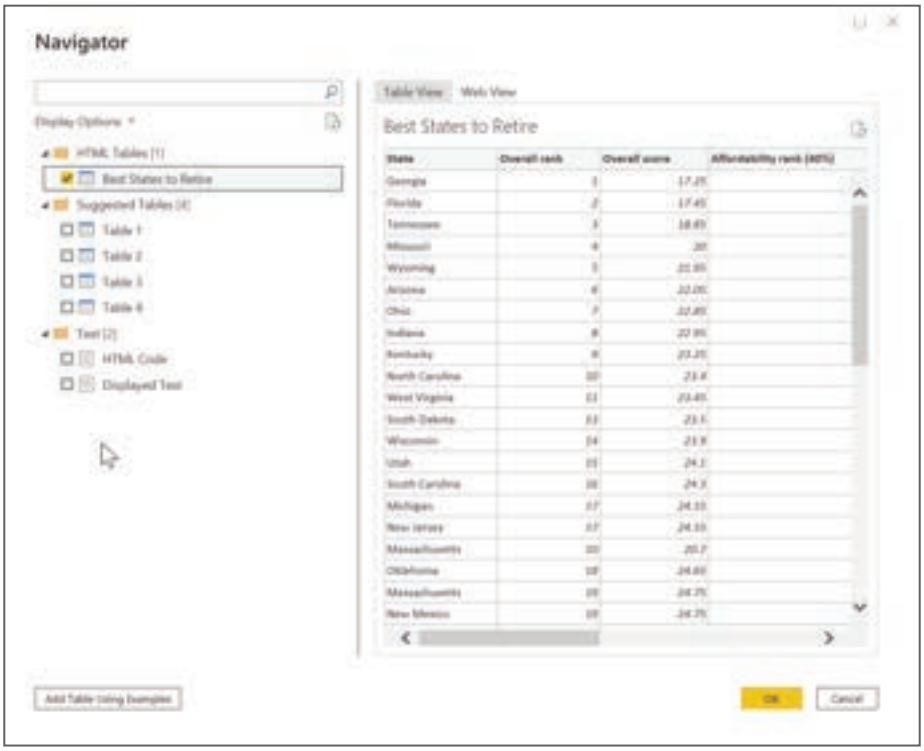


Figure 5.1.24.: The Navigator window

At this point, you can edit the query before loading the table, by selecting Transform data at the bottom of the window. Or, you can just load the table.

When you select Transform data, Power Query Editor starts, and a representative view of the table is shown. The Query Settings pane appears (if it doesn't, select the View tab on the ribbon, then select Show > Query Settings). Here's what it looks like.

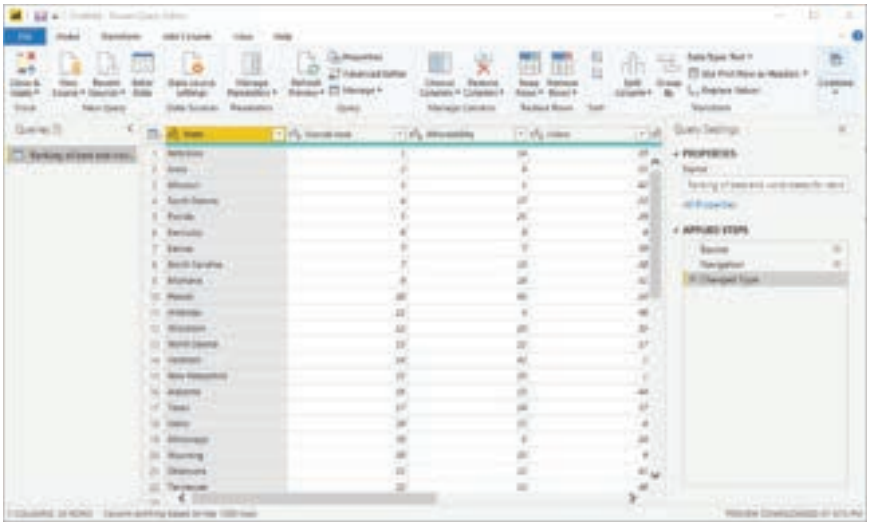


Figure 5.1.25.: Power Query Editor window

In Power BI Desktop, you can connect to multiple data sources and combine them to do interesting things.

In the next unit, we'll adjust the data to make it meet our needs. The process of adjusting connected data is called shaping.

SUMMARY

Power BI Desktop is a versatile tool for data analysis and reporting. It integrates Microsoft Power Query for Excel, data modeling, and visualization capabilities, working seamlessly with the online Power BI service.

Users follow a three-step process: connecting to data from multiple sources, shaping data through queries to create models, and generating reports. These reports can be saved as .pbix files and easily shared.

The tool offers three main views: Report view for visualizations, Data view for managing data, and Model view for relationships.

Power BI Desktop includes Power Query Editor for query building and data transformation.

The interface includes tabs like Home, Transform, Add Column, View, Tools, and Help.

Users can save their work through Close & Apply. It can connect to various data sources, as demonstrated by a web data connection example.

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 What is Power BI Desktop primarily used for?
   a) Web browsing
   b) Data analysis and reporting
   c) Video editing
   d) Gaming

2 Which Microsoft technology is integrated into Power BI Desktop for data transformation?
   a) Microsoft Word
   b) Microsoft Power Query for Excel
   c) Microsoft PowerPoint
   d) Microsoft Access

3 How can you easily share Power BI Desktop reports with others?
   a) Send them as email attachments
   b) Upload them to the Power BI service
   c) Share them on social media
   d) Print them as PDFs

4 Which of the following is NOT one of the primary views in Power BI Desktop?
   a) Report view
   b) Data view
   c) Query view
   d) Model view

5 In Power BI Desktop, what is the purpose of the Model view?
   a) Creating visualizations
   b) Data modeling and relationships management
   c) Data transformation
   d) Report sharing

6 Which tab in Power Query Editor allows you to perform common data transformation tasks?
   a) Home
   b) Transform
   c) View
   d) Tools

7 What is the tool in Power BI Desktop used to view and edit the code generated for each step in data transformation?
   a) Code Editor
   b) Power Query Editor
   c) Advanced Analyzer
   d) Query Inspector

8 How can you save your work in Power BI Desktop once you've finished shaping your data?
   a) By selecting "Exit"
   b) By clicking "Save As"
   c) By pressing Ctrl+S
   d) By choosing "Close & Apply"

9 Which file format is used to save Power BI Desktop reports?
   a) .xlsx
   b) .pptx
   c) .pbix
   d) .pdf

10 What is the first step when working with Power BI Desktop?
   a) Creating reports
   b) Shaping data
   c) Connecting to data
   d) Sharing reports

11 Which view in Power BI Desktop allows users to create compelling visualizations for reports?
   a) Data view
   b) Model view
   c) Report view
   d) Query view

12 In Power BI Desktop, what does the Transform tab offer access to?
   a) Data modeling
   b) Data transformation tasks
   c) Visualizations
   d) Report sharing

13 Which pane in Power BI Desktop displays the number of active queries and their names?
   a) Left (Queries) pane
   b) Center (Data) pane
   c) Right (Query Settings) pane
   d) Bottom (Advanced Editor) pane

14 What happens when you select a right-click menu item in Power BI Desktop's Query Settings pane?
   a) It changes the underlying data.
   b) It saves the step as part of the query.
   c) It deletes the entire query.
   d) It closes Power BI Desktop.

15 Which tab in Power Query Editor lets you see and edit the code generated for each step in data transformation?
   a) Code View
   b) Query Editor
   c) Advanced Editor
   d) Script Editor

Answers

1 *B. Data analysis and reporting*
2 *B. Microsoft Power Query for Excel*
3 *B. Upload them to the Power BI service*
4 *C. Query view*
5 *B. Data modeling and relationships management*
6 *B. Transform*
7 *B. Power Query Editor*
8 *D. By choosing "Close & Apply"*
9 *C. .pbix*
10 *C. Connecting to data*
11 *C. Report view*
12 *B. Data transformation tasks*
13 *A. Left (Queries) pane*
14 *B. It saves the step as part of the query.*
15 *C. Advanced Editor*

SELF-EXAMINATION QUESTIONS FOR PRACTICE:

1 What key Microsoft technology is integrated into Power BI Desktop for data transformation?

2 In Power BI Desktop, what is the primary purpose of the Model view?

3 Which tab in Power Query Editor provides access to common data transformation tasks?

4 What is the purpose of the Advanced Editor in Power Query Editor?

5 How can you save your work in Power BI Desktop once you've finished shaping your data?

6 What is the file format used to save Power BI Desktop reports?

7 What is the first step when working with Power BI Desktop?

8 In Power BI Desktop, which pane displays the number of active queries and their names?

9 What happens when you select a right-click menu item in Power BI Desktop's Query Settings pane?

10 Which tab in Power Query Editor lets you see and edit the code generated for each step in data transformation?

CHAPTER 5

5.2. COMBINE FILES (BINARIES) IN POWER BI DESKTOP

LEARNING OBJECTIVES

By the end of this module, you will be proficient in importing and consolidating multiple files with the same schema into a single logical table in Power BI Desktop. They will achieve the following key competencies:

- Folder Access: Understand how to connect to a folder, enter the folder path, and access files.
- Combining Files: Master the process of combining binary files using the Combine Files feature, including selecting content and applying transformations.
- Combine Files Behavior: Grasp how this feature analyzes file formats, creates exemplar and function queries, and extracts data.
- Efficient Data Consolidation: Learn to consolidate files within a folder with consistent formats and structures, and efficiently apply transformations.
- Folder Connection: Differentiate between connecting to a folder in Power Query Desktop and Power Query Online, and know how to select authentication methods.
- Troubleshooting: Develop skills to troubleshoot issues, ensuring files have consistent formats and structures, and optionally filtering files before combining.

Here's a powerful approach to importing data into Power BI Desktop: If you have multiple files that have the same schema, combine them into a single logical table. This popular technique has been made more convenient and more expansive.

To start the process of combining files from the same folder, select Get data, choose File > Folder, and then select Connect.

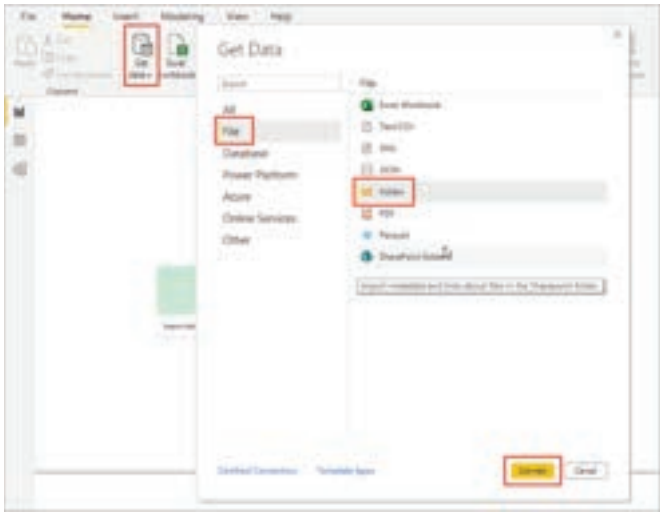


Figure 5.2.1.: Get Data dialog box highlighting the choice to connect to a folder.

Enter the folder path, select OK, and then choose Transform data to see the folder's files in Power Query Editor.

5.2.1. COMBINE FILES BEHAVIOR:

To combine binary files in Power Query Editor, select Content (the first column label) and choose Home > Combine Files. Or you can just select the Combine Files icon next to Content.

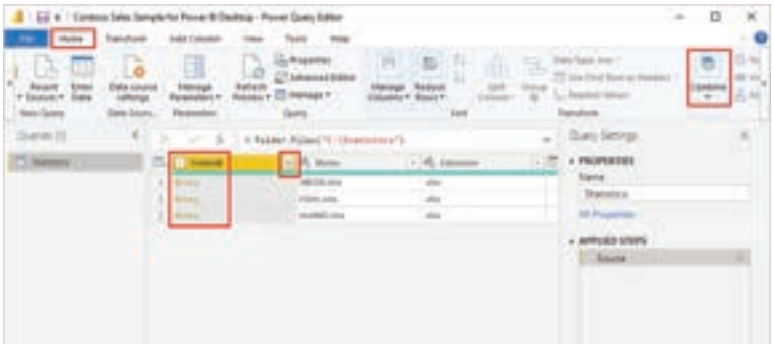


Figure 5.2.2.: Query Editor highlighting the icon to combine files.

The combine files transform behaves as follows:

- The combine files transform analyzes each input file to determine the correct file format to use, such as text, Excel workbook, or JSON file.
- The transform allows you to select a specific object from the first file, such as an Excel workbook, to extract.

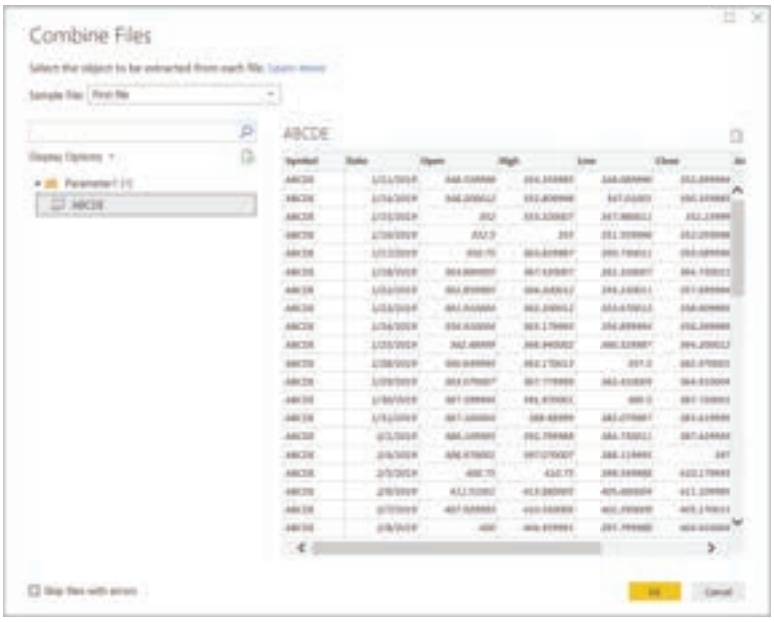


Figure 5.2.3.: Combine Files dialog box in Power BI desktop.

The combine files transform then automatically takes these actions:

- Creates an example query that performs all the required extraction steps in a single file.
- Creates a function query that parameterizes the file/binary input to the exemplar query. The exemplar query and the function query are linked, so that changes to the exemplar query are reflected in the function query.
- Applies the function query to the original query with input binaries, such as the Folder query. It applies the function query for binary inputs on each row, then expands the resulting data extraction as top-level columns.

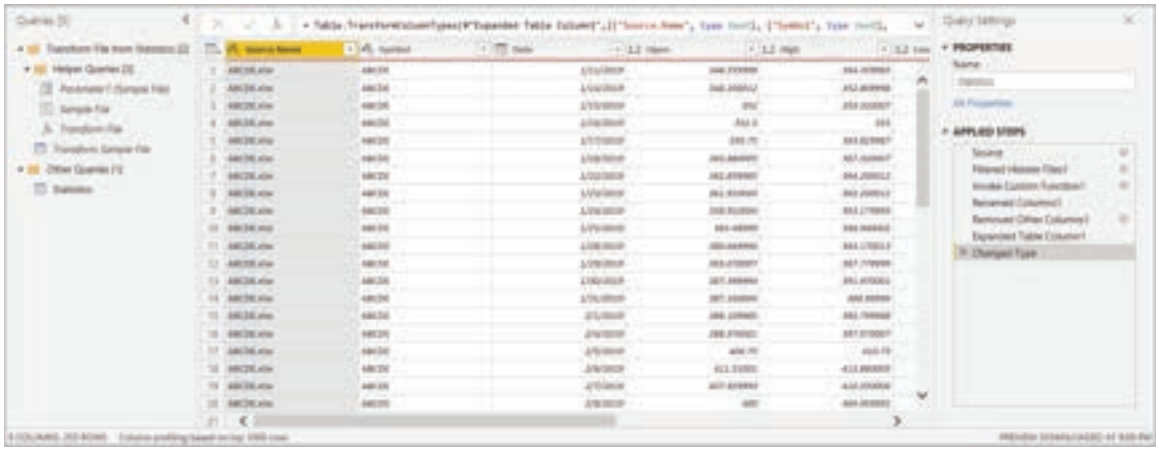


Figure 5.2.4.: The results of the combine files transform.

With the behavior of combine files, you can easily combine all files within a given folder if they have the same file type and structure (such as the same columns).

Also you can easily apply more transformation or extraction steps by modifying the automatically created exemplar query, without having to worry about modifying or creating other function query steps. Any changes to the exemplar query are automatically generated in the linked function query.

5.2.2. FOLDER

A Connect to a folder from Power Query Desktop:

To connect to a folder from Power Query Desktop:

- Select the Folder option in the connector selection.
- Enter the path to the folder you want to load, or select Browse to browse to the folder you want to load. Then select OK.

Figure 5.2.5.: Folder selection.

When you select the folder you want to use, the file information about all of the files in that folder are displayed. Also, file information about any files in any subfolders is also displayed.

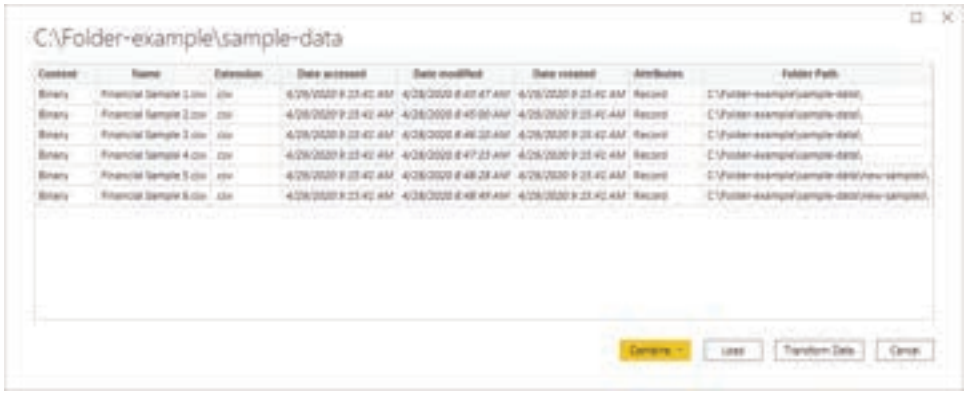


Figure 5.2.6.: Folder information.

- Select Combine & Transform Data to combine the data in the files of the selected folder and load the data in the Power Query Editor for editing. Select Combine & Load to load the data from all of the files in the folder directly into your app. Or select Transform Data to load the folder data as-is in the Power Query Editor.

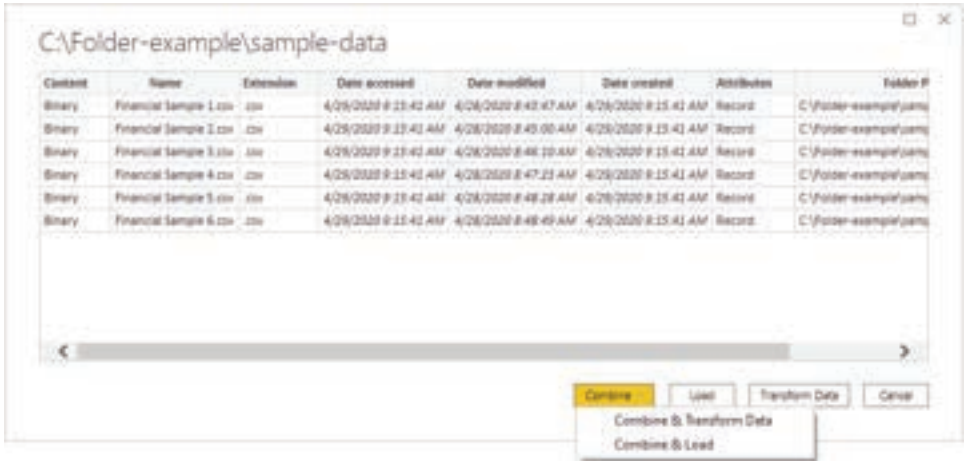


Figure 5.2.7.: Combine files from folder.

B Connect to a folder from Power Query Online

To connect to a folder from Power Query Online:

- Select the Folder option in the connector selection.
- Enter the path to the folder you want to load.

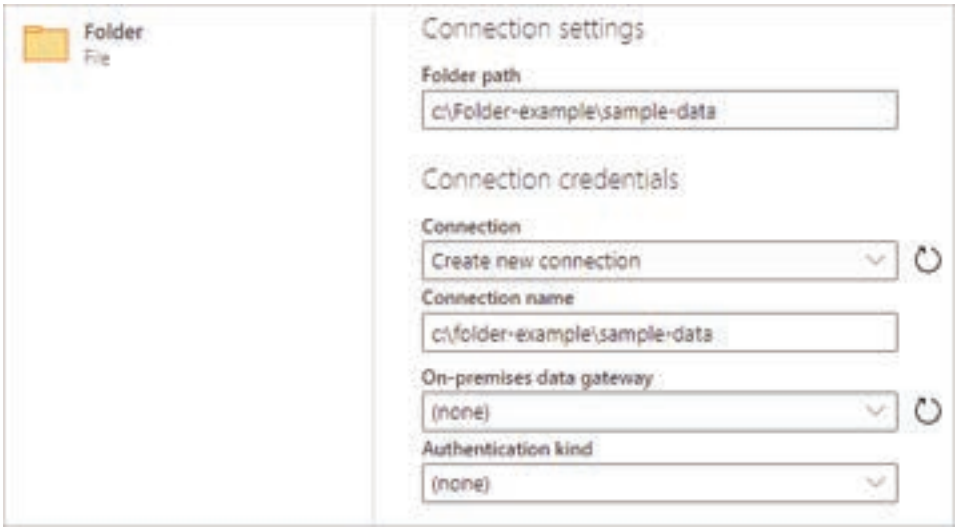


Figure 5.2.8.: Folder selection online.

- Enter the name of an on-premises data gateway that you'll use to access the folder.
- Select the authentication kind to connect to the folder. If you select the Windows authentication kind, enter your credentials.
- Select Next.
- In the Navigator dialog box, select Combine to combine the data in the files of the selected folder and load the data into the Power Query Editor for editing. Or select Transform data to load the folder data as-is in the Power Query Editor.

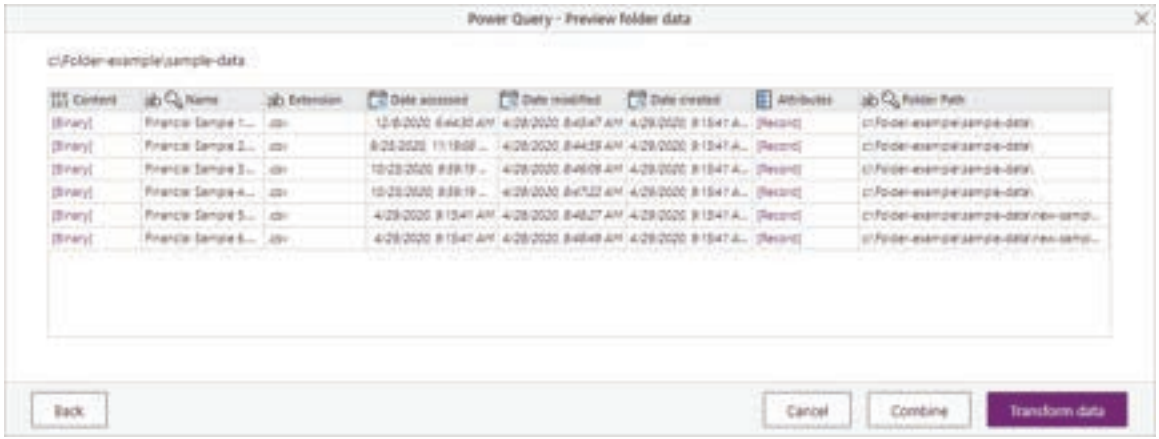


Figure 5.2.9.: Select what to do with the data displayed in the Navigator.

5.2.3. TROUBLESHOOTING:

Combining files

When you combine files using the folder connector, all the files in the folder and its subfolders are processed the same way, and the results are then combined. The way the files are processed is determined by the example file you select. For example, if you select an Excel file and choose a table called "Table1", then all the files will be treated as Excel files that contain a table called "Table1".

To ensure that combining the files works properly, make sure that all the files in the folder and its subfolders have the same file format and structure. If you need to exclude some of the files, first select Transform data instead of Combine and filter the table of files in the Power Query Editor before combining.

SUMMARY

- Access Files: Begin by connecting to a folder via "Get data" > "File > Folder" in Power Query Editor.
- Combine Files: Use the "Combine Files" feature to automatically analyze and extract data from binary files, simplifying the consolidation process.
- Folder Connection: Learn how to connect to folders, view file information, and select loading options in Power Query Desktop or Online.
- Troubleshooting: Ensure consistent file formats and structures for successful file combination, or filter files if needed before combining.

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 What is the primary purpose of combining multiple files with the same schema in Power BI Desktop?
   a) To reduce file storage space
   b) To improve data visualization
   c) To create a single logical table
   d) To generate data backups

2 How can you access the Combine Files feature in Power Query Editor?
   a) By clicking "Save As"
   b) By selecting "Home > Combine Files"
   c) By using the "Print" option
   d) By clicking "Refresh All"

3 In the Combine Files feature, what does the "Content" column represent?
   a) File names
   b) File size
   c) File content type
   d) File date of creation

4 What is the purpose of the exemplar query in the Combine Files transformation?
   a) To delete unwanted files
   b) To select specific files to combine
   c) To perform data extraction steps in a single file
   d) To apply advanced transformations

5 Which query type is linked to the exemplar query in the Combine Files transformation?
   a) Function query
   b) Subquery
   c) Conditional query
   d) Aggregate query

6 When can you easily combine all files within a folder using the Combine Files feature in Power BI Desktop?
   a) When files have different structures
   b) When files have different content types
   c) When files share the same file type and structure
   d) When files have different file names

7 What does the Folder connector in Power Query Desktop allow you to do?
   a) Create new folders
   b) Delete folders
   c) Connect to a folder and access its files
   d) Rename folders

8 In Power Query Online, what is the purpose of selecting an on-premises data gateway when connecting to a folder?
   a) To download additional files
   b) To enable cloud storage
   c) To secure the folder connection
   d) To access online folders

9 What should you ensure when using the folder connector for combining files to avoid issues?
   a) Consistent file formats and structures
   b) Varying file content types
   c) Different file names
   d) Complex folder hierarchies

10 What action should you take if you need to exclude specific files when using the folder connector for combining?
   a) Delete the files physically
   b) Use the "Transform data" option
   c) Rename the files
   d) Change the file format

11 What is the function of the Combine Files icon next to the "Content" column in Power Query Editor?
   a) To delete files
   b) To add new files
   c) To initiate file combination
   d) To change file formats

12 When combining files in Power Query Editor, what is the purpose of the linked exemplar and function queries?
   a) To merge queries
   b) To create a backup of the data
   c) To apply advanced transformations
   d) To perform data extraction and ensure changes propagate

13 In Power Query Desktop, what is the outcome of selecting "Combine & Load" when connecting to a folder?
   a) The files are combined, but not loaded into the Power Query Editor
   b) The files are combined and loaded directly into the app
   c) The files are loaded as-is without combining
   d) The files are deleted

14 What is the primary advantage of connecting to a folder using the Folder connector in Power Query Desktop?
   a) Faster data extraction
   b) Greater data security
   c) Simplified file management
   d) Access to cloud-based files

15 When should you use the "Transform data" option instead of "Combine" in the folder connector to ensure specific files are excluded?
   a) When you want to delete the files
   b) When you need to perform advanced transformations
   c) When you want to load the files directly into the app
   d) When you want to filter and exclude files before combining

Answers

1 *C) To create a single logical table*
2 *B) By selecting "Home > Combine Files"*
3 *C) File content type*
4 *C) To perform data extraction steps in a single file*
5 *A) Function query*
6 *C) When files share the same file type and structure*
7 *C) Connect to a folder and access its files*
8 *C) To secure the folder connection*
9 *A) Consistent file formats and structures*
10 *B) Use the "Transform data" option*
11 *C) To initiate file combination*
12 *D) To perform data extraction and ensure changes propagate*
13 *B) The files are combined and loaded directly into the app*
14 *C) Simplified file management*
15 *D) When you want to filter and exclude files before combining*

SELF-EXAMINATION QUESTIONS FOR PRACTICE:

1 What is the primary advantage of combining multiple files with the same schema into a single logical table in Power BI Desktop?

2 How do you initiate the process of combining files from the same folder in Power BI Desktop?

3 In the Combine Files feature, what does the "Content" column represent, and why is it important?

4 Describe the behavior of the Combine Files transform in Power Query Editor.

5 What are the automatic actions performed by the Combine Files transform when consolidating data from multiple binary files?

6 When can you easily combine all files within a folder using the Combine Files feature in Power BI Desktop?

7 Explain the significance of the exemplar query and the function query in the context of combining files.

8 Differentiate between connecting to a folder from Power Query Desktop and Power Query Online. What are the key considerations when selecting one over the other?

9 What steps should you take to troubleshoot issues when combining files using the folder connector in Power BI Desktop?

10 Why is it crucial for all files in a folder to have consistent file formats and structures when using the folder connector for file combination in Power BI Desktop?

CHAPTER 5

5.3. GROUPING AND AGGREGATION:

LEARNING OBJECTIVES

After completing this training module on Power Query, you will be able to:
Demonstrate proficiency in using Power Query's Group by feature

- Identify the locations of the Group by button in Power Query.
- Differentiate between two types of grouping operations: Column groupings and Row groupings.
- Apply aggregate functions to group data by one or more columns.
- Perform operations at both the row level and the column level using Group by.
- Extract specific information from grouped data, such as finding top-performing products.
- Understand and implement fuzzy grouping techniques for approximate text matching.
- Utilize various options within fuzzy grouping, including similarity thresholds, case sensitivity, and transformation tables.

In Power Query, you can group values in various rows into a single value by grouping the rows according to the values in one or more columns. You can choose from two types of grouping operations:

- Column groupings.
- Row groupings.

	Year	Country	Product	Sales Channel	Units
1	2020	USA	Shirt	Online	5000
2	2020	USA	Shorts	Online	4000
3	2020	USA	Shirt	Reseller	7500
4	2020	USA	Shorts	Reseller	4500
5	2020	Panama	Shirt	Online	55
6	2020	Panama	Shorts	Online	70
7	2020	Panama	Shirt	Reseller	200
8	2020	Panama	Shorts	Reseller	150
9	2020	Canada	Shirt	Online	1200
10	2020	Canada	Shorts	Online	1450
11	2020	Canada	Shirt	Reseller	700
12	2020	Canada	Shorts	Reseller	800

Figure 5.3.1.: Sample initial table.

5.3.1. WHERE TO FIND THE GROUP BY BUTTON:

You can find the Group by button in three places:

On the Home tab, in the Transform group.

Figure 5.3.2.: Group by on the Home tab.

On the Transform tab, in the Table group.

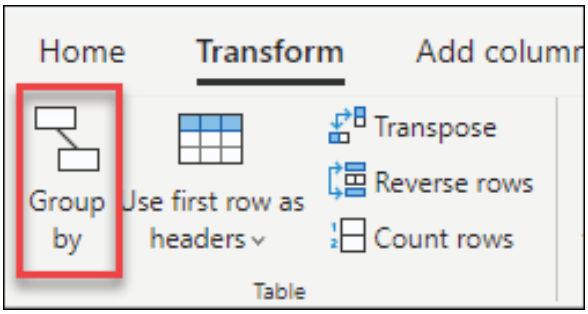


Figure 5.3.3.: Group by on the Transform tab.

On the shortcut menu when you right-click to select columns.

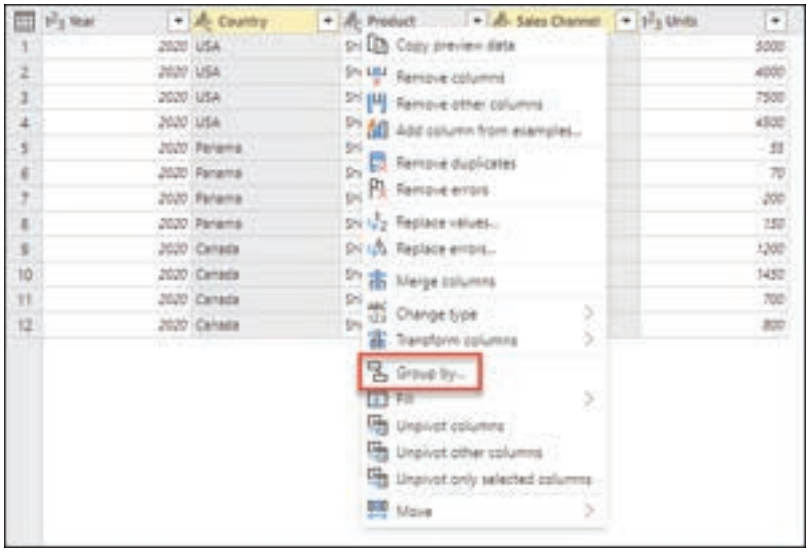


Figure 5.3.4.: Group by on the shortcut menu.

5.3.2. USE AN AGGREGATE FUNCTION TO GROUP BY ONE OR MORE COLUMNS:

In this example, your goal is to summarize the total units sold at the country and sales channel level. You'll use the Country and Sales Channel columns to perform the group by operation.

- Select Group by on the Home tab.
- Select the Advanced option, so you can select multiple columns to group by.
- Select the Country column.
- Select Add grouping.
- Select the Sales Channel column.
- In New column name, enter Total units, in Operation, select Sum, and in Column, select Units.
- Select OK

Figure 5.3.5.: Group by dialog box with aggregated columns.

This operation gives you the following table.

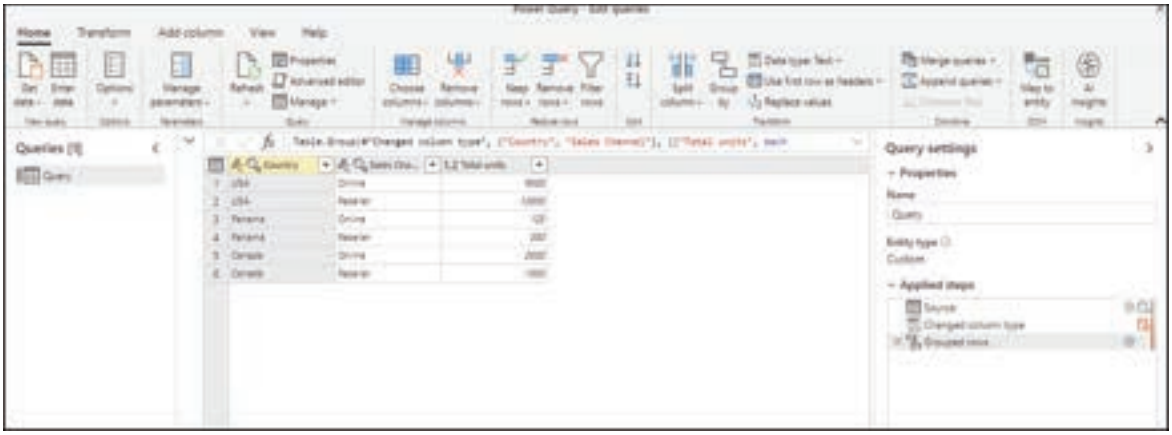


Figure 5.3.6.: Sample output table with Country, Sales Channel, and Total units columns.

5.3.3. OPERATIONS AVAILABLE:

With the Group by feature, the available operations can be categorized in two ways:

- Row level operation
- Column level operation

The following table describes each of these operations.

Operation Name	Category	Description
Sum	Column operation	Sums up all values from a column
Average	Column operation	Calculates the average value from a column
Median	Column operation	Calculates the median from a column
Min	Column operation	Calculates the minimum value from a column
Max	Column operation	Calculates the maximum value from a column
Percentile	Column operation	Calculates the percentile, using an input value from 0 to 100, from a column
Count distinct values	Column operation	Calculates the number of distinct values from a column
Count rows	Row operation	Calculates the total number of rows from a given group
Count distinct rows	Row operation	Calculates the number of distinct rows from a given group
All rows	Row operation	Outputs all grouped rows in a table value with no aggregations

Figure 5.3.7.: Operations

5.3.4. PERFORM AN OPERATION TO GROUP BY ONE OR MORE COLUMNS:

Starting from the original sample, in this example you'll create a column containing the total units and two other columns that give you the name and units sold for the top-performing product, summarized at the country and sales channel level.

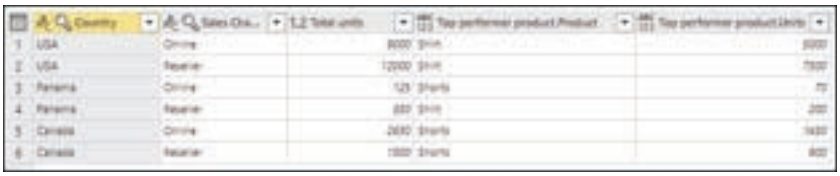


Figure 5.3.8.: Sample output table with operations.

1 Use the following columns as Group by columns:

- Country
- xSales Channel

2 Create two new columns by following these steps:

- Aggregate the Units column by using the Sum operation. Name this column Total units.
- Add a new Products column by using the All rows operation.

Figure 5.3.9.: Group by dialog box with a non-aggregate column.

After that operation is complete, notice how the Products column has [Table] values inside each cell. Each [Table] value contains all the rows that were grouped by the Country and Sales Channel columns from your original table. You can select the white space inside the cell to see a preview of the contents of the table at the bottom of the dialog box.

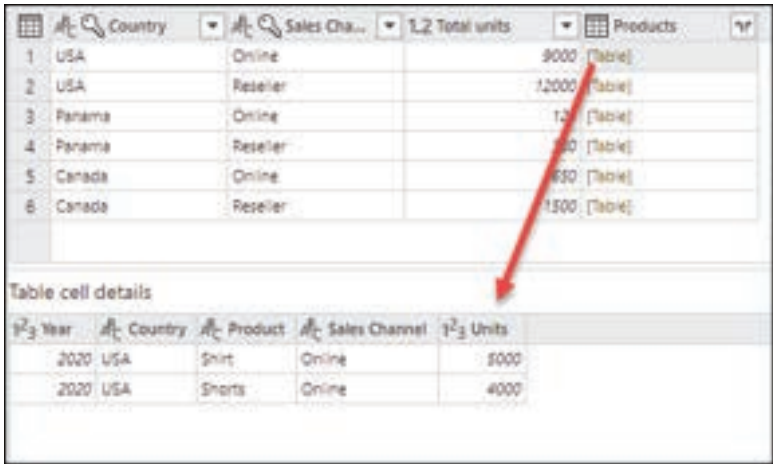


Figure 5.3.10.: Table details preview pane.

Next, you need to extract the row that has the highest value in the Units column of the tables inside the new Products column, and call that new column Top performer product.

5.3.5. EXTRACT THE TOP PERFORMER PRODUCT INFORMATION:

With the new Products column with [Table] values, you create a new custom column by going to the Add Column tab on the ribbon and selecting Custom column from the General group.

Figure 5.3.11.: Add a custom column.

Name your new column Top performer product. Enter the formula Table.Max ([Products], "Units") under Custom column formula.

Figure 5.3.12.: Custom column formula with Table.Max.

The result of that formula creates a new column with [Record] values. These record values are essentially a table with just one row. These records contain the row with the maximum value for the Units column of each [Table] value in the Products column.

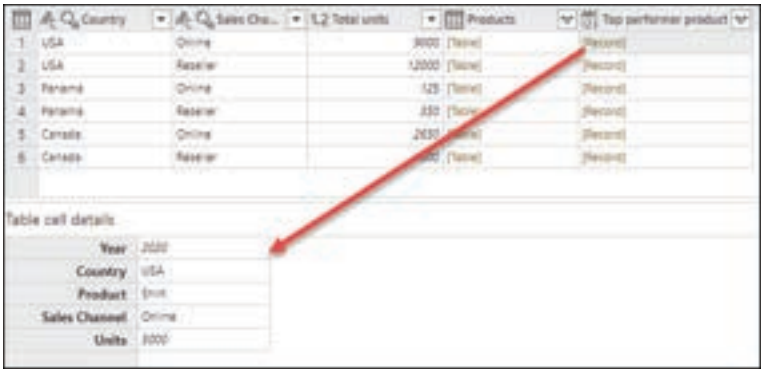


Figure 5.3.13.: Result of the custom column formula with Table.Max.

With this new Top performer product column that contains [Record] values, you can select the expand. expand icon, select the Product and Units fields, and then select OK.

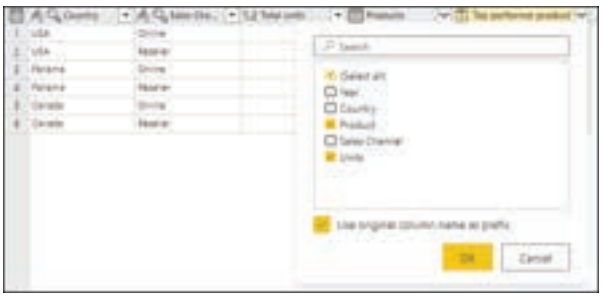


Figure 5.3.14.: Expand operation for record value on the Top performer product column.

After removing your Products column and setting the data type for both newly expanded columns, your result will resemble the following image.

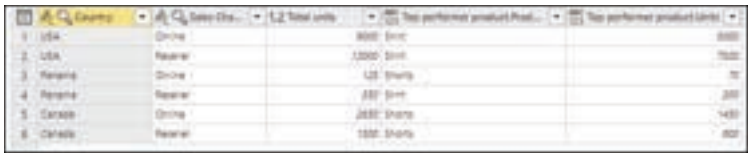


Figure 5.3.15.: Final table with all transformations.

5.3.6. FUZZY GROUPING:

To demonstrate how to do "fuzzy grouping," consider the sample table shown in the following image.

	1²3 id	Person
1	1	miguel
2	2	Miguel
3	3	migueeel
4	4	mike
5	5	Mike
6	6	William
7	7	Bill
8	8	billy
9	9	Miguel

Figure 5.3.16.: Table with nine rows of entries that contain various spellings and capitalizations of the name Miguel and William.

The goal of fuzzy grouping is to do a group-by operation that uses an approximate match algorithm for text strings. Power Query uses the Jaccard similarity algorithm to measure the similarity between pairs of instances. Then it applies agglomerative hierarchical clustering to group instances together. The following image shows the output that you expect, where the table will be grouped by the Person column.

	Person	Frequency
1	Miguel	4
2	mike	2
3	Bill	2
4	William	1

Figure 5.3.17.: Table showing entries for Person as "Miguel" and "Mike," and Frequency as 3 and 2, respectively."

To do the fuzzy grouping, you perform the same steps previously described in this article. The only difference is that this time, in the Group by dialog box, you select the Use fuzzy grouping check box.

Figure 5.3.18.: Fuzzy grouping check box in the Group by dialog box.

For each group of rows, Power Query will pick the most frequent instance as the "canonical" instance. If multiple instances occur with the same frequency, Power Query will pick the first one. After you select OK in the Group by dialog box, you'll get the result that you were expecting.

	Person	Frequency
1	Miguel	4
2	mike	2
3	Bill	2
4	William	1

Figure 5.3.19.: Fuzzy grouping sample final table, no transform table.

However, you have more control over the fuzzy grouping operation by expanding Fuzzy group options.

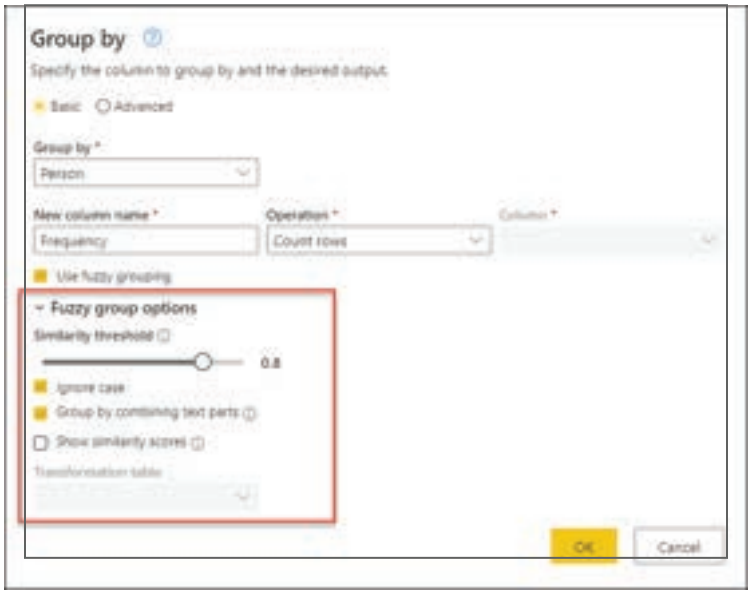


Figure 5.3.20.: Fuzzy group options.

The following options are available for fuzzy grouping:

- Similarity threshold (optional): This option indicates how similar two values must be to be grouped together. The minimum setting of 0 will cause all values to be grouped together. The maximum setting of 1 will only allow values that match exactly to be grouped together. The default is 0.8.
- Ignore case: When comparing text strings, case will be ignored. This option is enabled by default.
- Group by combining text parts: The algorithm will try to combine text parts (such as combining Micro and soft into Microsoft) to group values.
- Show similarity scores: Show similarity scores between the input values and the computed representative values after fuzzy grouping. Requires the addition of an operation such as All rows to showcase this information on a row-by-row level.
- Transformation table (optional): You can select a transformation table that will map values (such as mapping MSFT to Microsoft) to group them together.

For this example, a transformation table will be used to demonstrate how values can be mapped. The transformation table has two columns:

- From: The text string to look for in your table.
- To: The text string to use to replace the text string in the From column.

The following image shows the transformation table used in this example.

	From	To
1	mike	Miguel
2	William	Bill

Figure 5.3.21.: Table showing From values of mike and William, and To values of Miguel and Bill.

Important:

It's important that the transformation table has a the same columns and column names as shown above (they have to be "From" and "To"), otherwise Power Query will not recognize these.

Return to the Group by dialog box, expand Fuzzy group options, change the operation from Count rows to All rows, enable the Show similarity scores option, and then select the Transformation table drop-down menu.

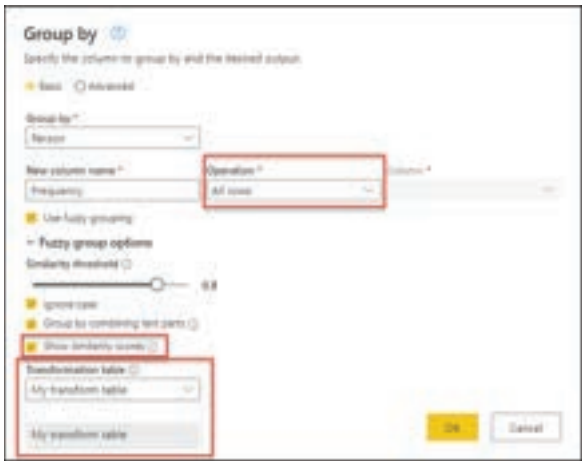
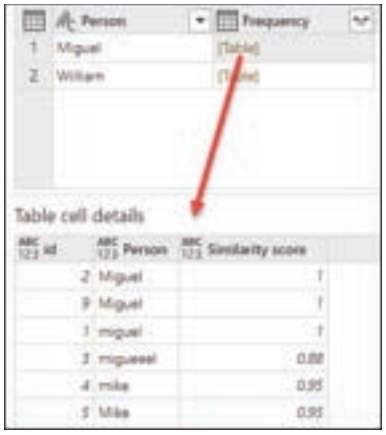


Figure 5.3.22.:Fuzzy grouping sample transformation table drop-down menu.



After you select the transformation table, select OK. The result of that operation gives you the following information:

Figure 5.3.23.: Fuzzy grouping sample final table with transform table.

In this example, the Ignore case option was enabled, so the values in the From column of the Transformation table are used to look for the text string without considering the case of the string. This transformation operation occurs first, and then the fuzzy grouping operation is performed.

The similarity score is also shown in the table value next to the person column, which reflects exactly how the values were grouped and their respective similarity scores. You can expand this column if needed or use the values from the new Frequency columns for other sorts of transformations.

SUMMARY

- Grouping Data: Understanding how to group rows based on column values, with options for column and row groupings. Users learn to locate the Group by button in three places, including the Home and Transform tabs, and on the shortcut menu.
- Aggregate Functions: Practical applications of aggregate functions to summarize data, illustrated through an example of totalling units sold by country and sales channel.
- Operations Overview: An overview of row-level and column-level operations in Power Query's Group by feature, with a comprehensive table describing each operation.
- Advanced Transformations: Demonstrating how to create new columns based on grouped data, extracting valuable insights like the top-performing product.
- Fuzzy Grouping: Introducing the concept of fuzzy grouping, enabling approximate text matching using the Jaccard similarity algorithm and hierarchical clustering. Options for similarity thresholds, case sensitivity, and transformation tables are discussed.

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 What are the two primary types of grouping operations available in Power Query?
   a) Row grouping and Page grouping
   b) Column grouping and Row grouping
   c) Filter grouping and Sort grouping
   d) Aggregate grouping and Join grouping

2 Where can you find the Group by button in Power Query?
   a) Only on the Home tab
   b) Only on the Transform tab
   c) Only on the shortcut menu
   d) In three places: Home tab, Transform tab, and shortcut menu

3 Which option allows you to select multiple columns for grouping in Power Query?
   a) Basic option
   b) Standard option
   c) Advanced option
   d) Premium option

4 To summarize the total units sold at the country and sales channel level, which aggregate function should you use in Power Query?
   a) Average
   b) Count
   c) Sum
   d) Maximum

5 In the context of Power Query, what do "row level operation" and "column level operation" refer to?
   a) The order in which operations are applied
   b) The type of data source being used
   c) Two categories of operations in Group by
   d) The level of user access

6 How can you extract the top-performing product from grouped data in Power Query?
   a) By using the Filter operation
   b) By creating a custom column with the Table.Max function
   c) By sorting the data by product name
   d) By using the Remove Duplicates operation

7 What is the primary goal of fuzzy grouping in Power Query?
a) To perform exact text matching
b) To group data based on exact matches
c) To handle approximate text matching
d) To count the number of rows in a table

8 What is the Jaccard similarity algorithm used for in Power Query's fuzzy grouping?
a) Measuring the similarity between pairs of text strings
b) Sorting data alphabetically
c) Calculating the sum of numeric values
d) Grouping data based on exact matches

9 In fuzzy grouping, what does the "similarity threshold" control?
a) The maximum number of rows in a group
b) How similar two values must be to be grouped together
c) The total number of transformations applied
d) The minimum number of columns in a table

10 What does the "Ignore case" option do in fuzzy grouping?
a) Ignores rows with missing values
b) Treats text strings as case-sensitive
c) Ignores letter case when comparing text strings
d) Excludes numeric columns from grouping

11 What is the purpose of the "Group by combining text parts" option in fuzzy grouping?
a) It combines numeric values within a group.
b) It merges text parts to form new group names.
c) It separates grouped data into multiple tables.
d) It creates subgroups within a larger group.

12 What does the "Show similarity scores" option provide in fuzzy grouping results?
a) It displays the total number of groups created.
b) It shows a visual representation of the data.
c) It presents similarity scores between input values and computed representative values.
d) It calculates the average similarity score of the grouped data.

13 What is the role of a transformation table in fuzzy grouping?
a) It defines the order of operations in fuzzy grouping.
b) It specifies which columns to exclude from grouping.
c) It maps values to replace text strings, helping to group similar entries.
d) It stores data for reference purposes.

14 What is the minimum requirement for the column names in a transformation table used for fuzzy grouping in Power Query?
   a) They must be different from the original column names.
   b) They must be in uppercase letters only.
   c) They must match the original column names ("From" and "To").
   d) They must be in reverse order compared to the original column names.

15 What happens when you enable the "Use fuzzy grouping" option in Power Query's Group by dialog box?
   a) It triggers an exact text matching operation.
   b) It initiates a hierarchical clustering process.
   c) It activates numeric calculations.
   d) It enables case sensitivity.

Answers

1 *b) Column grouping and Row grouping*
2 *d) In three places: Home tab, Transform tab, and shortcut menu*
3 *c) Advanced option*
4 *c) Sum*
5 *c) Two categories of operations in Group by*
6 *b) By creating a custom column with the Table.Max function*
7 *c) To handle approximate text matching*
8 *a) Measuring the similarity between pairs of text strings*
9 *b) How similar two values must be to be grouped together*
10 *c) Ignores letter case when comparing text strings*
11 *b) It merges text parts to form new group names.*
12 *c) It presents similarity scores between input values and computed representative values.*
13 *c) It maps values to replace text strings, helping to group similar entries.*
14 *c) They must match the original column names ("From" and "To").*
15 *b) It initiates a hierarchical clustering process.*

SELF-EXAMINATION QUESTIONS FOR PRACTICE:

1 What are the two primary types of grouping operations available in Power Query, and how do they differ from each other?

2 In Power Query, where can you locate the Group by button, and what are the three places mentioned in the content?

3 Describe the steps involved in using an aggregate function to group data by one or more columns in Power Query.

4 How can you create a new column that totals units sold by country and sales channel using Power Query?

5 What are the two categories into which the available operations in the Group by feature can be classified, as mentioned in the content?

6 Explain the process of creating a custom column to extract the top-performing product from grouped data in Power Query.

7 What is the purpose of fuzzy grouping in Power Query, and how does it handle approximate text matching?

8 Mention at least two options available when configuring fuzzy grouping in Power Query and describe their significance.

9 What is the importance of a transformation table in fuzzy grouping, and what are the key columns it should contain?

10 How does Power Query handle variations in text data when performing fuzzy grouping, and what information is typically displayed to aid in understanding the grouping process?

CHAPTER 5

5.4. PARAMETERS AND CUSTOM FUNCTIONS:

If you find yourself in a situation where you need to apply the same set of transformations to different queries or values, creating a Power Query custom function that can be reused as many times as you need could be beneficial. A Power Query custom function is a mapping from a set of input values to a single output value, and is created from native M functions and operators.

While you can manually create your own Power Query custom function using code, the Power Query user interface offers you features to speed up, simplify, and enhance the process of creating and managing a custom function.

LEARNING OBJECTIVES

- Understanding Custom Functions: Gain an understanding of Power Query custom functions, which allow for the creation of reusable data transformation processes using native M functions and operators.
- Creating Custom Functions: Learn how to create custom functions from table references, such as folders with multiple files, and comprehend the steps involved.
- Applying Transformations: Master the process of applying transformations to sample queries, including interpreting binary data, removing unnecessary rows, and promoting headers.
- Invoking Custom Functions: Discover how to invoke custom functions to automate transformations for multiple rows or values.
- Adding Parameters: Learn how to add parameters to existing custom functions and manage their impact on associated queries.
- Reusable Logic: Explore the creation of custom functions from reusable logic, facilitating efficient data transformation across various queries or values.
- Practical Benefits: Understand how custom functions enhance productivity by simplifying and automating complex data preparation tasks in Power Query.

Create a custom function from a table reference:

You can follow along with this example by downloading the sample files used in this article from the following download link. For simplicity, this article will be using the Folder connector. To learn more about the Folder connector, go to Folder. The goal of this example is to create a custom function that can be applied to all the files in that folder before combining all of the data from all files into a single table.

Start by using the Folder connector experience to navigate to the folder where your files are located and select Transform Data or Edit. This will take you to the Power Query experience.

Right-click on the Binary value of your choice from the Content field and select the Add as New Query option. For this example, you'll see that the selection was made for the first file from the list, which happens to be the file April 2019.csv.

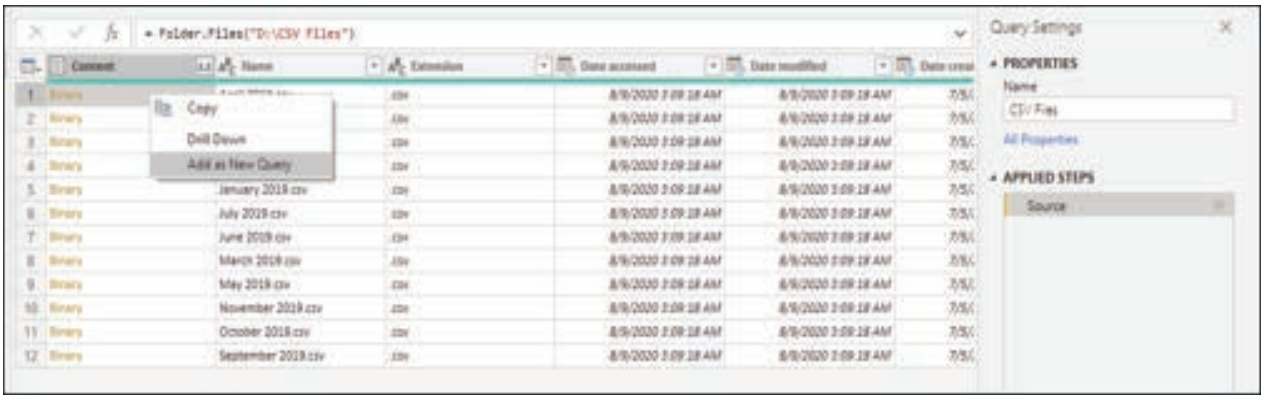


Figure 5.4.1.: Selecting a file to be our Sample File.

This option will effectively create a new query with a navigation step directly to that file as a Binary, and the name of this new query will be the file path of the selected file. Rename this query to be Sample File.

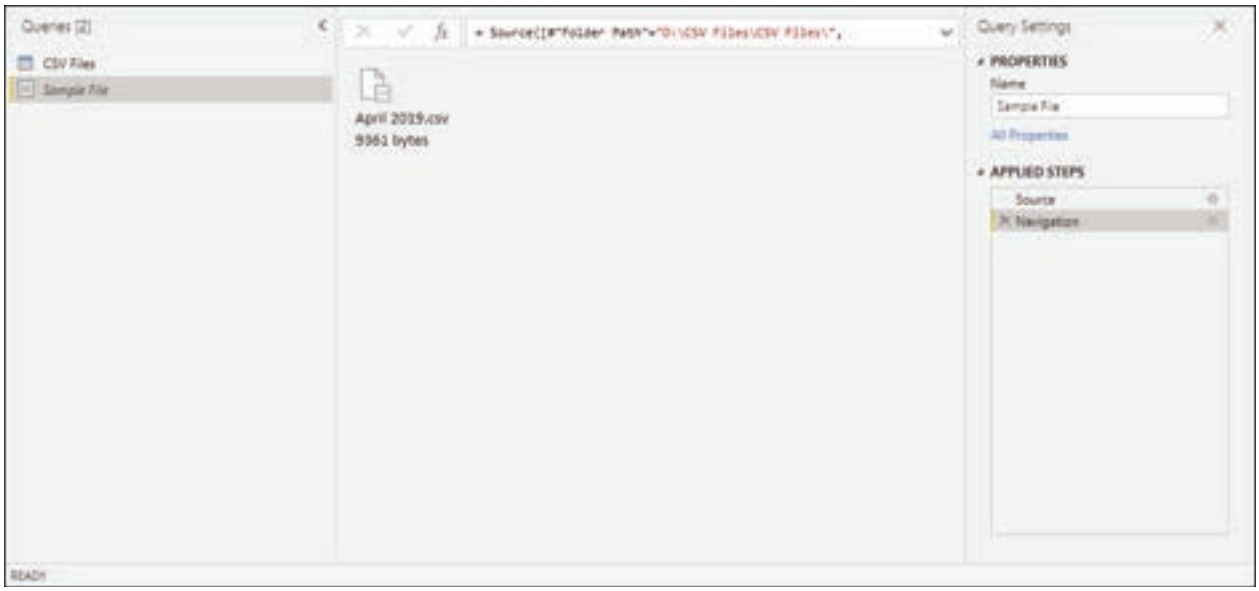


Figure 5.4.2.: Sample File query.

Create a new parameter with the name File Parameter. Use the Sample File query as the Current Value, as shown in the following image.

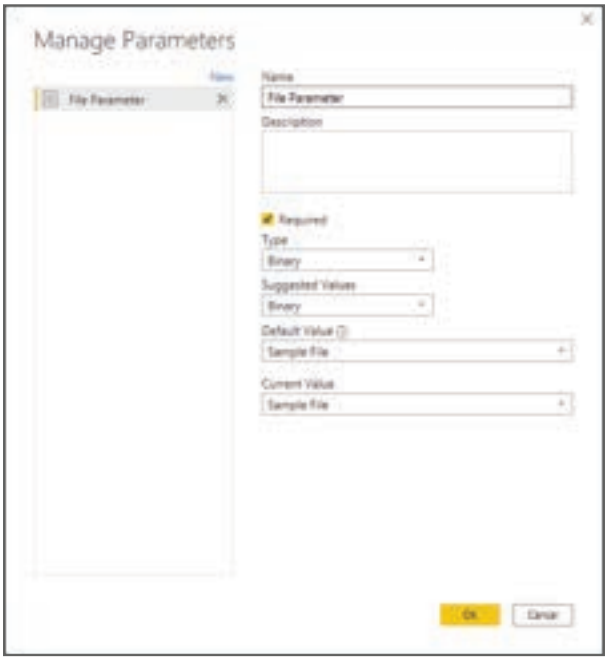


Figure 5.4.3.: File parameter.

Right-click File Parameter from the Queries pane. Select the Reference option.

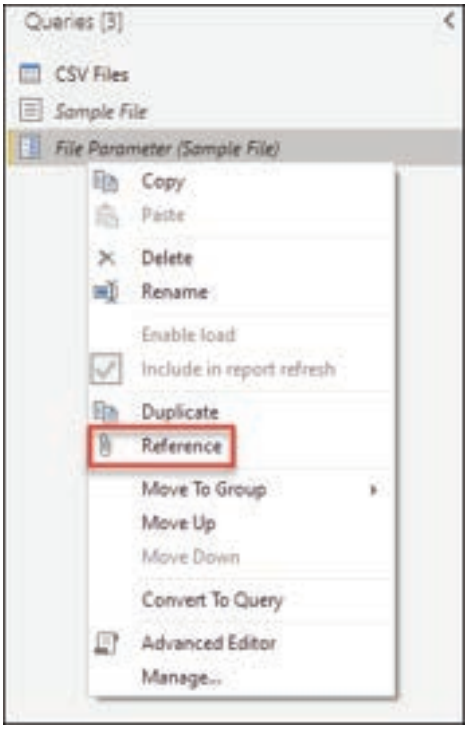


Figure 5.4.4.: Reference the File Parameter.

Rename the newly created query from File Parameter (2) to Transform Sample file.

Figure 5.4.5.: Renamed query Transform Sample file.

Right-click this new Transform Sample file query and select the Create Function option.

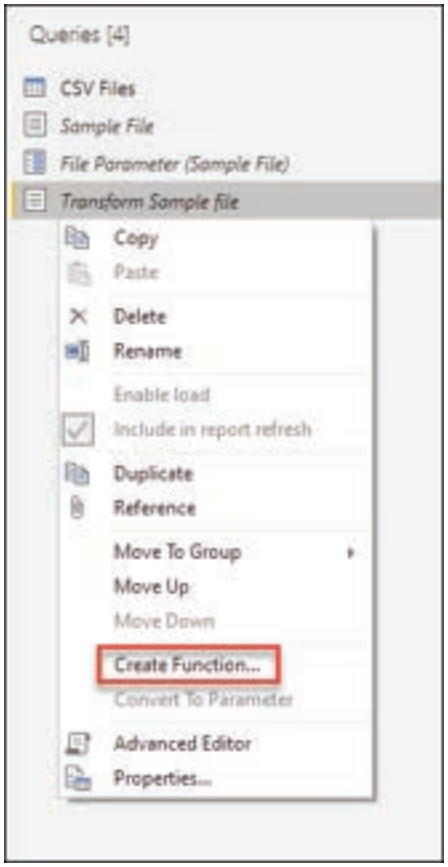


Figure 5.4.6.: Create a function from Transform Sample file.

This operation will effectively create a new function that will be linked with the Transform Sample file query. Any changes that you make to the Transform Sample file query will be automatically replicated to your custom function. During the creation of this new function, use Transform file as the Function name.

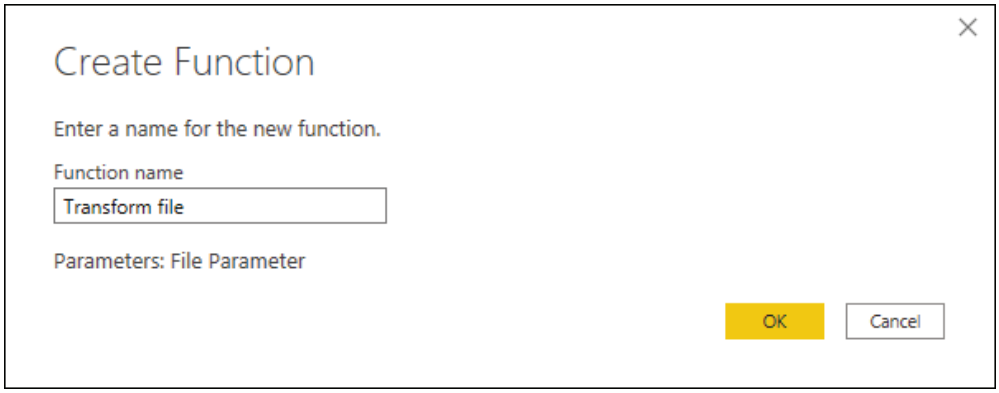


Figure 5.4.7.: Create a function window for Transform file.

After creating the function, you'll notice that a new group will be created for you with the name of your function. This new group will contain:After creating the function, you'll notice that a new group will be created for you with the name of your function. This new group will contain:

- All parameters that were referenced in your Transform Sample file query.
- Your Transform Sample file query, commonly known as the sample query.
- Your newly created function, in this case Transform file.

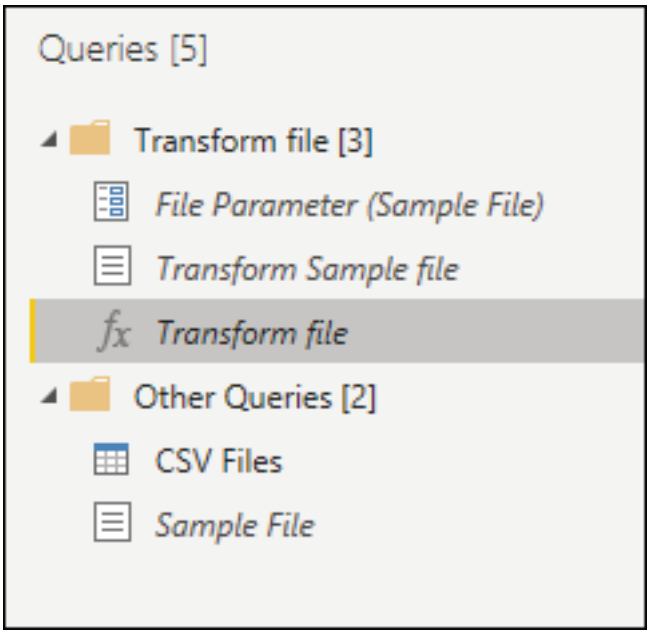


Figure 5.4.8.: Function group.

Applying transformations to a sample query:

With your new function created, select the query with the name Transform Sample file. This query is now linked with the Transform file function, so any changes made to this query will be reflected in the function. This is what is known as the concept of a sample query linked to a function.

The first transformation that needs to happen to this query is one that will interpret the binary. You can right-click the binary from the preview pane and select the CSV option to interpret the binary as a CSV file.

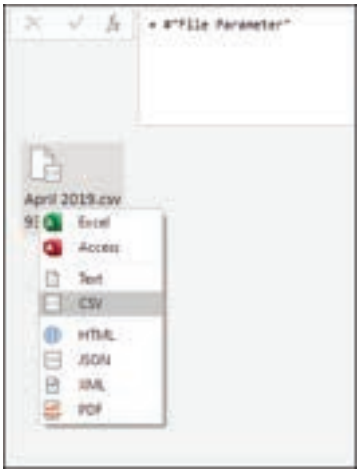


Figure 5.4.9.: Interpret binary as CSV.

The format of all the CSV files in the folder is the same. They all have a header that spans the first top four rows. The column headers are located in row five and the data starts from row six downwards, as shown in the next image.

	Column1	Column2	Column3	Column4
1	Report generated on 01-01-2020			
2	Created by: user9284			
3	Company XYZ			
4				
5	Date	Country	Units	Revenue
6	2019-04-22	Brazil	153	2649.32
7	2019-04-14	Brazil	57	940.4
8	2019-04-26	Colombia	310	4408.22
9	2019-04-25	USA	90	2044.18
10	2019-04-23	Panama	204	16330.85
11	2019-04-07	USA	356	3772.26
12	2019-04-11	Colombia	122	2490.96
13	2019-04-27	Colombia	367	19933.19
14	2019-04-24	Panama	223	13834.04
15	2019-04-16	Colombia	159	3448.16
16	2019-04-08	Canada	258	14601.34
17	2019-04-14	Panama	325	11959.47
18	2019-04-01	Colombia	349	10844.36
19	2019-04-07	Panama	139	2890.93

Figure 5.4.10.: Sample CSV data.

The next set of transformation steps that need to be applied to the Transform Sample file are:

1 Remove the top four rows—This action will get rid of the rows that are considered part of the header section of the file.

Figure 5.4.11.: Remove top rows from Transform Sample file.

2 Promote headers—The headers for your final table are now in the first row of the table. You can promote them as shown in the next image.

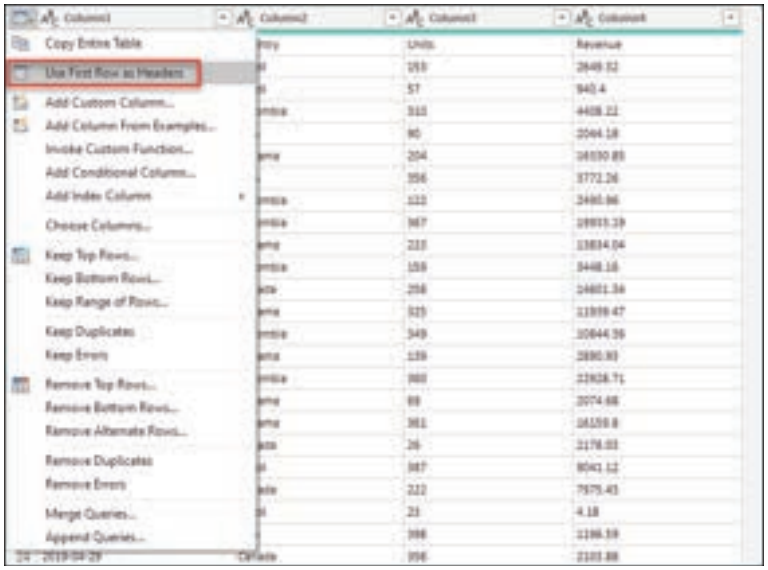


Figure 5.4.12.: Use first row as headers.

Power Query by default will automatically add a new Changed Type step after promoting your column headers that will automatically detect the data types for each column. Your Transform Sample file query will look like the next image.

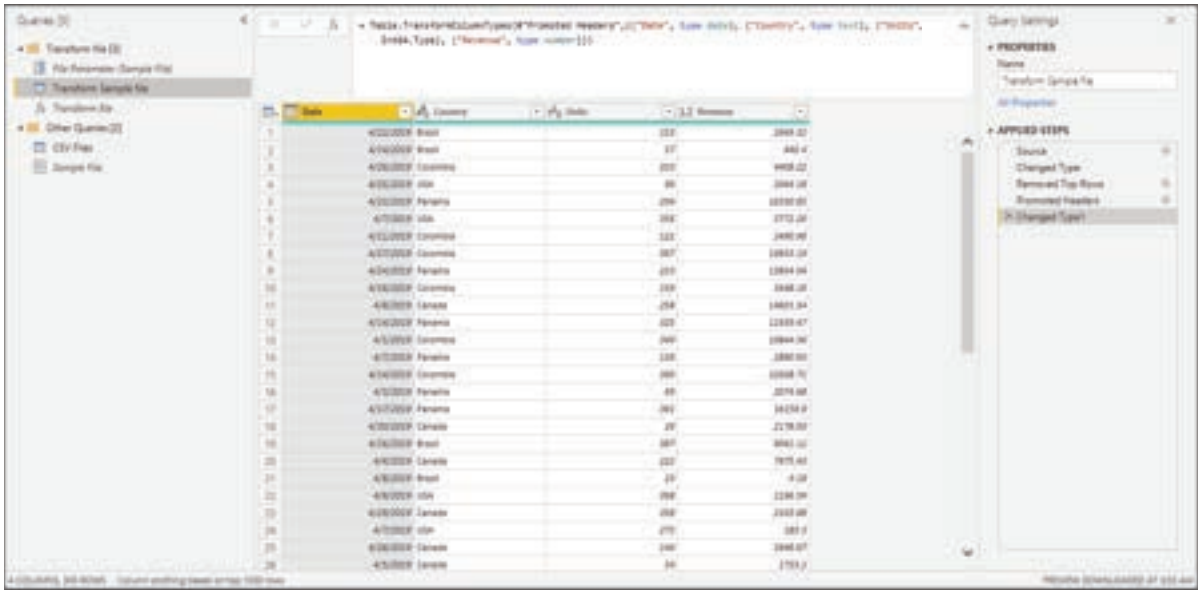


Figure 5.4.13.: Final transform sample query.

Caution:

Your Transform file function relies on the steps performed in the Transform Sample file query. However, if you try to manually modify the code for the Transform file function, you'll be greeted with a warning that reads The definition of the function 'Transform file' is updated whenever query 'Transform Sample file' is updated. However, updates will stop if you directly modify function 'Transform file'.

Invoke a custom function as a new column:

With the custom function now created and all the transformation steps incorporated, you can go back to the original query where you have the list of files from the folder. Inside the Add Column tab in the ribbon, select Invoke Custom Function from the General group. Inside the Invoke Custom Function window, enter Output Table as the New column name. Select the name of your function, Transform file, from the Function query dropdown. After selecting the function from the dropdown menu, the parameter for the function will be displayed and you can select which column from the table to use as the argument for this function. Select the Content column as the value / argument to be passed for the File Parameter.

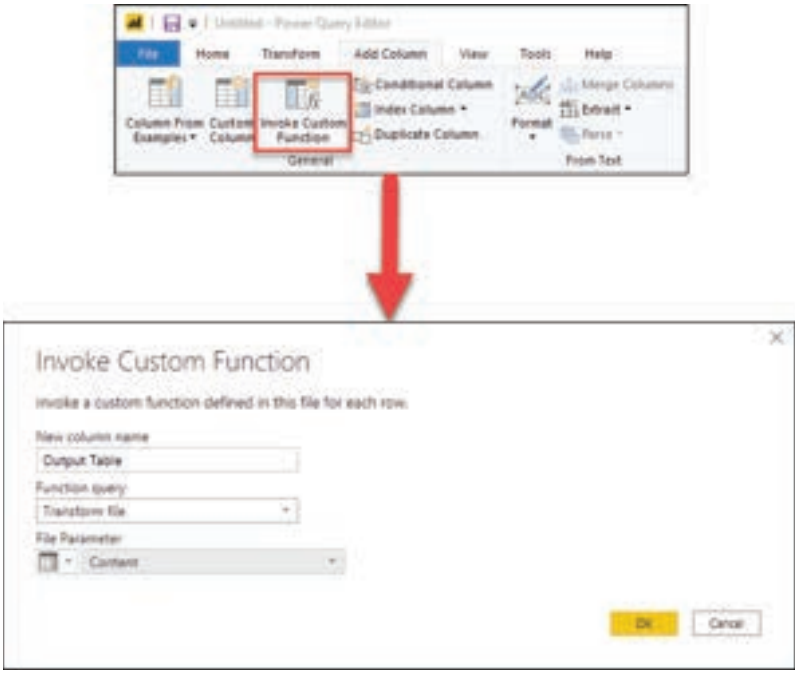


Figure 5.4.14.: Invoke custom function button in Add column menu.

After you select OK, a new column with the name Output Table will be created. This column has Table values in its cells, as shown in the next image. For simplicity, remove all columns from this table except Name and Output Table.

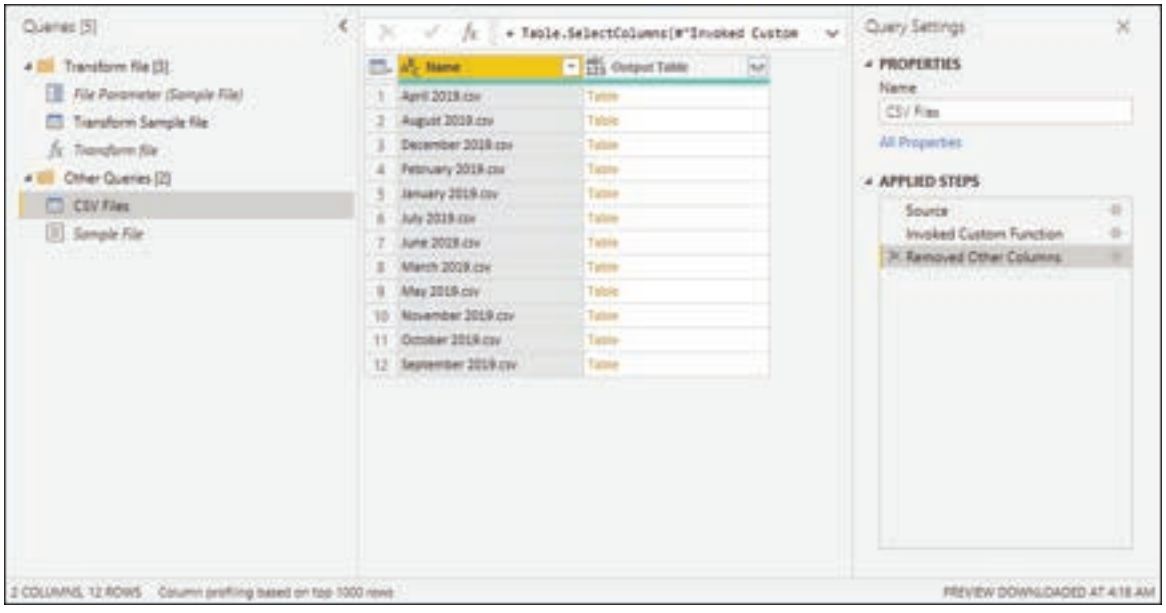


Figure 5.4.15: Custom function invoked.

Your function was applied to every single row from the table using the values from the Content column as the argument for your function. Now that the data has been transformed into the shape that you're looking for, you can expand the Output Table column, as shown in the image below, without using any prefix for the expanded columns.

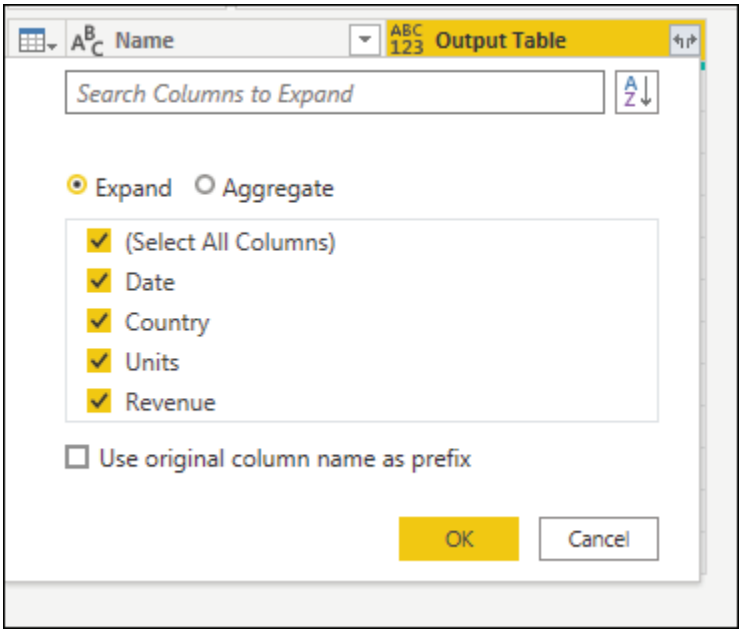


Figure 5.4.16.: Expand Output Table column.

You can verify that you have data from all files in the folder by checking the values in the Name or Date column. For this case, you can check the values from the Date column, as each file only contains data for a single month from a given year. If you see more than one, it means that you've successfully combined data from multiple files into a single table.

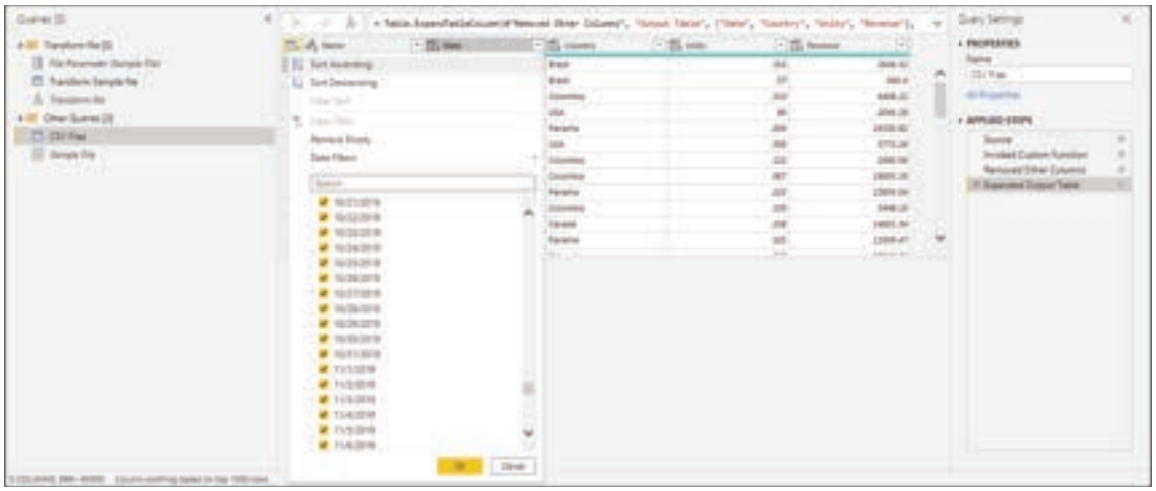


Figure 5.4.17.: Validating that the final table contains data from all files.

Add new parameter to existing custom function:

Imagine that there's a new requirement on top of what you've built. The new requirement requires that before you combine the files, you filter the data inside them to only get the rows where the Country is equals to Panama.

To make this requirement happen, create a new parameter called Market with the text data type. For the Current Value, enter the value Panama.

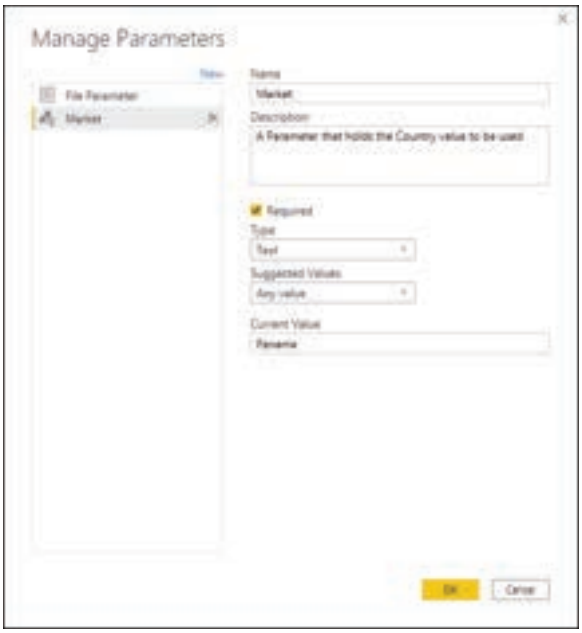


Figure 5.4.18.: New parameter.

With this new parameter, select the Transform Sample file query and filter the Country field using the value from the Market parameter.

Figure 5.4.19.: Filter Country column using the new Market parameter.

Applying this new step to your query will automatically update the Transform file function, which will now require two parameters based on the two parameters that your Transform Sample file uses.

Figure 5.4.20.: Function updated with now two parameters.

But the CSV files query has a warning sign next to it. Now that your function has been updated, it requires two parameters. So the step where you invoke the function results in error values, since only one of the arguments was passed to the Transform file function during the Invoked Custom Function step.

Figure 5.4.21.: Errors after function update.

To fix the errors, double-click Invoked Custom Function in the Applied Steps to open the Invoke Custom Function window. In the Market parameter, manually enter the value Panama.

Figure 5.4.22.: Updating Invoked Custom Function arguments.

You can now check your query to validate that only rows where Country is equal to Panama show up in the final result set of the CSV Files query.

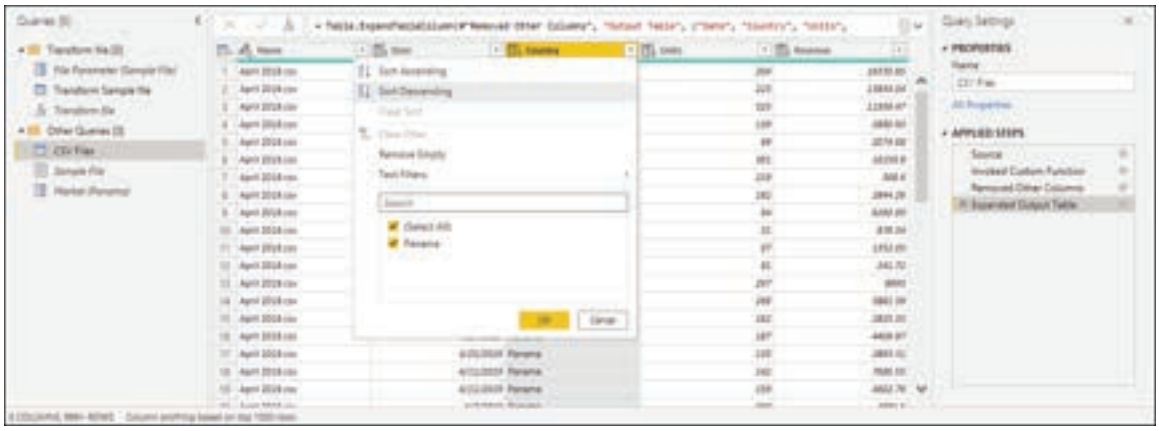


Figure 5.4.23.: Final output table after updated arguments.

Create a custom function from a reusable piece of logic:

If you have multiple queries or values that require the same set of transformations, you could create a custom function that acts as a reusable piece of logic. Later, this custom function can be invoked against the queries or values of your choice. This custom function could save you time and help you in managing your set of transformations in a central location, which you can modify at any moment.

For example, imagine a query that has several codes as a text string and you want to create a function that will decode those values, as in the following sample table:

code
PTY-CM1090-LAX
LAX-CM701-PTY
PTY-CM4441-MIA
MIA-UA1257-LAX
LAX-XY2842-MIA

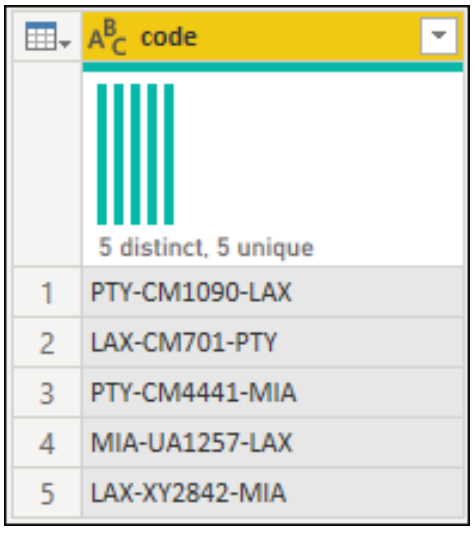


Figure 5.4.24.: List of codes.

You start by having a parameter that has a value that serves as an example. For this case, it will be the value PTY-CM1090-LAX.

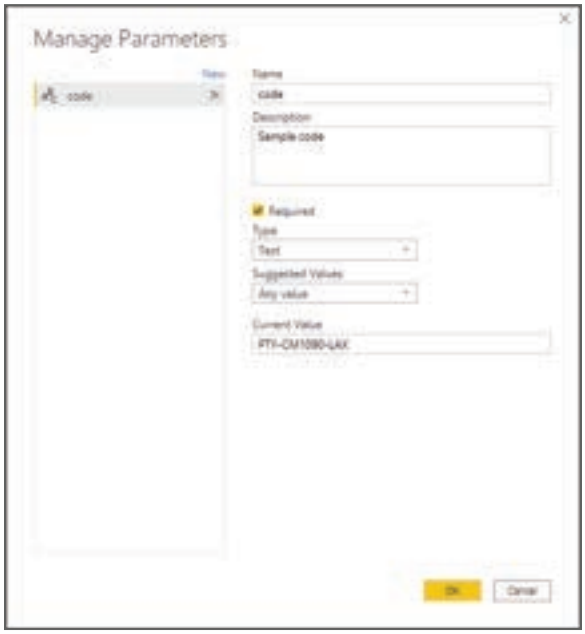


Figure 5.4.25.: Sample parameter code value.

From that parameter, you create a new query where you apply the transformations that you need. For this case, you want to split the code PTY-CM1090-LAX into multiple components:

Origin = PTY
Destination = LAX
Airline = CM
FlightID = 1090

Figure 5.4.26.: Sample transform query.

The M code for that set of transformations is shown below.

Power Query M

```
let
  Source = code,
  SplitValues = Text.Split( Source, "-"),
    CreateRow = [Origin= SplitValues{0}, Destination= SplitValues{2}, Airline=Text.Start( SplitValues{1},2), FlightID= Text.End( SplitValues{1}, Text.Length( SplitValues{1} ) - 2) ],
  RowToTable = Table.FromRecords( { CreateRow } ),
    #"Changed Type" = Table.TransformColumnTypes(RowToTable,{{"Origin", type text}, {"Destination", type text}, {"Airline", type text}, {"FlightID", type text}})
in
  #"Changed Type"
```

You can then transform that query into a function by doing a right-click on the query and selecting Create Function. Finally, you can invoke your custom function into any of your queries or values, as shown in the next image.

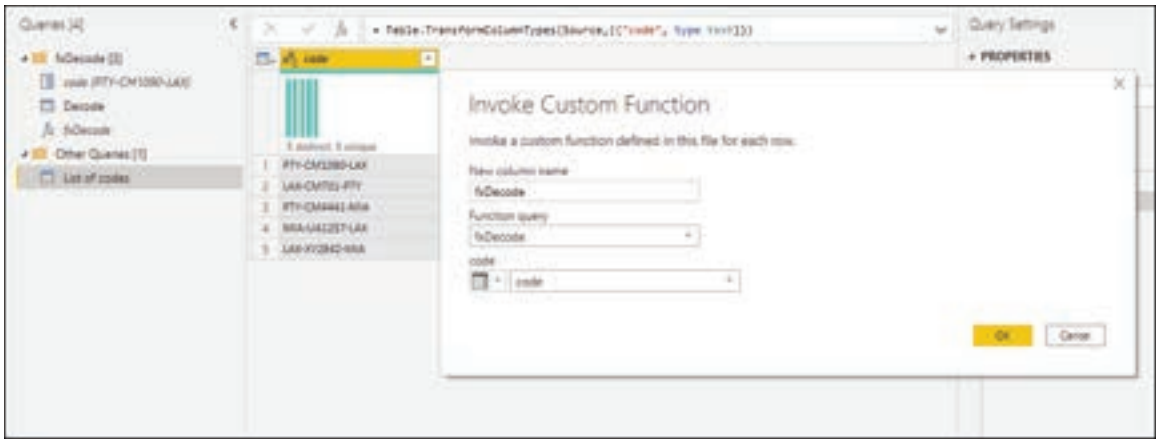


Figure 5.4.27.: Invoking a custom function.

After a few more transformations, you can see that you've reached your desired output and leveraged the logic for such a transformation from a custom function.

Figure 5.4.28.: Final output query after invoking a custom function.

SUMMARY

This section explains how to create custom functions in Power Query using table references:

Initial Setup:

- Select a folder with multiple files.
- Create a query from one file's binary data.
- Rename it as "Sample File."

Parameter and Function:

- Create a "File Parameter."
- Reference it as "Transform Sample file."
- Create a custom function, "Transform file."

Transformations:

- Apply transformations to "Transform Sample file," like interpreting binary as CSV and data cleanup.

Invoking Custom Function:

- Use the custom function to consolidate data from multiple files into a new column.
- Validate data consolidation.

Adding Parameters:

- Add a new parameter, e.g., "Market," and use it to filter data.
- Manually update the custom function parameters when there are errors.

Reusable Functions:

- Create a custom function for reusable data transformations, such as decoding text values.

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 What is the primary purpose of creating custom functions in Power Query?
   a) To design custom user interfaces
   b) To automate data extraction from websites
   c) To encapsulate and reuse data transformation logic
   d) To create interactive data visualizations

2 Which connector is used to initiate the process of creating a custom function in Power Query?
   a) Web
   b) Folder
   c) Database
   d) Excel

3 What should you rename a query created from the binary value of a file when setting up a custom function?
   a) Original Data
   b) File Parameter
   c) Transform Sample file
   d) Sample File

4 What is the relationship between a query and a custom function in Power Query?
   a) They are entirely independent of each other.
   b) Queries can only reference built-in functions, not custom ones.
   c) A query can be linked to a custom function, allowing updates to propagate.
   d) A query can only reference custom functions, not built-in ones.

5 What are some common initial transformations applied to a sample query when creating a custom function?
   a) Sorting and filtering
   b) Renaming columns
   c) Interpreting binary data as CSV
   d) Calculating averages

6 When updating a custom function, what happens if you manually modify the function's code?
   a) It results in an error.
   b) It creates a new function.
   c) It has no impact on the function.
   d) It enhances the function's performance.

7 How can you resolve parameter errors when updating a custom function in Power Query?
   a) Delete the custom function and recreate it.
   b) Ignore the errors; they won't affect the function.
   c) Manually enter missing parameter values.
   d) Contact Microsoft Support for assistance.

8 What is the advantage of creating reusable custom functions in Power Query?
   a) They improve query performance.
   b) They simplify the process of data extraction.
   c) They allow you to apply the same transformations to multiple queries.
   d) They enable real-time data updates.

9 In which scenario might creating a custom function for data transformation be useful?
   a) When working with static data
   b) When dealing with data stored in a single file
   c) When frequently repeating a set of data transformations
   d) When creating complex data visualizations

10 Which connector is used to access data from a web source in Power Query?
   a) Folder
   b) Web
   c) Excel
   d) Database

11 What does Power Query automatically add after promoting column headers in a data transformation?
   a) Sort step
   b) Filter step
   c) Changed Type step
   d) Aggregate step

12 In Power Query, what is the purpose of referencing a parameter from a query?
   a) It locks the parameter's value.
   b) It creates a new query.
   c) It ensures parameter consistency.
   d) It deletes the parameter.

13 What happens when you select "Invoke Custom Function" in Power Query?
   a) It creates a new custom function.
   b) It invokes all available functions.
   c) It adds a new column based on the custom function.
   d) It deletes the current query.

14 What is the primary benefit of using a custom function to encapsulate data transformation logic in Power Query?
   a) It simplifies data extraction from web sources.
   b) It makes query performance faster.
   c) It allows for centralized and reusable transformations.
   d) It enables real-time data updates.

15 Which query element displays a warning when its function requires multiple parameters?
   a) The original query
   b) The custom function
   c) The applied steps
   d) The parameter list

Answers

1 *C) To encapsulate and reuse data transformation logic*
2 *B) Folder*
3 *D) Sample File*
4 *C) A query can be linked to a custom function, allowing updates to propagate.*
5 *C) Interpreting binary data as CSV*
6 *A) It results in an error.*
7 *C) Manually enter missing parameter values.*
8 *C) They allow you to apply the same transformations to multiple queries.*
9 *C) When frequently repeating a set of data transformations*
10 *B) Web*
11 *C) Changed Type step*
12 *C) It ensures parameter consistency.*
13 *C) It adds a new column based on the custom function.*
14 *C) It allows for centralized and reusable transformations.*
15 *B) The custom function*

SELF-EXAMINATION QUESTIONS FOR PRACTICE:

1 What is the purpose of creating custom functions in Power Query?

2 How can you initiate the process of creating a custom function in Power Query?

3 Why is it important to start with a "Sample File" when creating custom functions?

4 How do you link a query with a custom function in Power Query?

5 What are some common data transformations applied to a sample query when creating a custom function?

6 What does it mean when Power Query displays a warning about updating the custom function?

7 How can you resolve parameter errors when updating a custom function in Power Query?

8 Why is it beneficial to create reusable custom functions in Power Query?

9 Can you provide an example of a scenario where creating a custom function for data transformation would be useful?

10 What are the key steps involved in creating a custom function from reusable logic in Power Query?

CHAPTER 5

5.5. ERROR HANDLING

LEARNING OBJECTIVES

- Distinguish between step-level and cell-level errors in Power Query.
- Identify the components of a step-level error, including error reason, error message, and error detail.
- Recognize common step-level errors, such as "Can't find the source" and "The column of the table wasn't found," and understand their causes.
- Implement appropriate solutions for common step-level errors.
- Understand how to handle cell-level errors in Power Query, including removing, replacing, or keeping errors.
- Identify common cell-level errors, such as data type conversion errors, operation errors, and nested value errors.
- Apply solutions to address common cell-level errors, including modifying data sources, changing data types, and adjusting privacy levels.

In Power Query, you can encounter two types of errors:

- Step-level errors
- Cell-level errors

This section provides suggestions for how to fix the most common errors you might find at each level, and describes the error reason, error message, and error detail for each.

5.5.1. STEP-LEVEL ERROR:

A step-level error prevents the query from loading and displays the error components in a yellow pane.

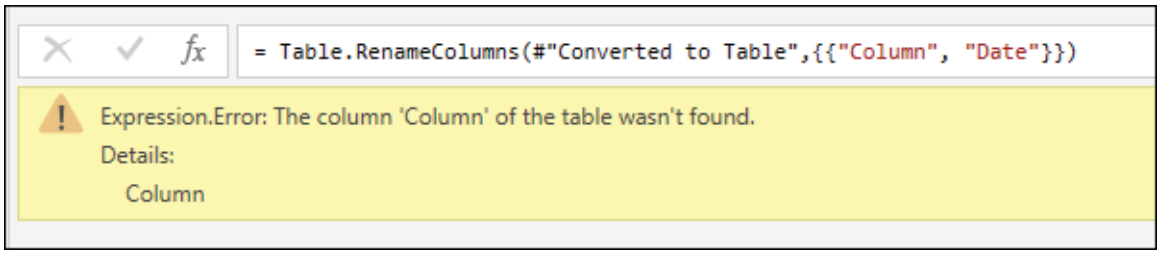


Figure 5.5.1.: Step-level error.

- Error reason: The first section before the colon. In the example above, the error reason is Expression.Error.

- Error message: The section directly after the reason. In the example above, the error message is The column 'Column' of the table wasn't found.
- Error detail: The section directly after the Details: string. In the example above, the error detail is Column.

Common step-level errors:

In all cases, we recommend that you take a close look at the error reason, error message, and error detail to understand what's causing the error. You can select the Go to error button, if available, to view the first step where the error occurred.

Figure 5.5.2.: Go to error button.

Can't find the source - DataSource.Error

This error commonly occurs when the data source is inaccessible by the user, the user doesn't have the correct credentials to access the data source, or the source has been moved to a different place.

Example: You have a query from a text tile that was located in drive D and created by user A. User A shares the query with user B, who doesn't have access to drive D. When this person tries to execute the query, they get a DataSource.Error because there's no drive D in their environment.

Figure 5.5.3.: Data source error, could not find the file because there's no drive D in the current environment.

Possible solutions: You can change the file path of the text file to a path that both users have access to. As user B, you can change the file path to be a local copy of the same text file. If the Edit settings button is available in the error pane, you can select it and change the file path.

The column of the table wasn't found:

This error is commonly triggered when a step makes a direct reference to a column name that doesn't exist in the query.

Example: You have a query from a text file where one of the column names was Column. In your query, you have a step that renames that column to Date. But there was a change in the original text file, and it no longer has a column heading with the name Column because it was manually changed to Date. Power Query is unable to find a column heading named Column, so it can't rename any columns. It displays the error shown in the following image.

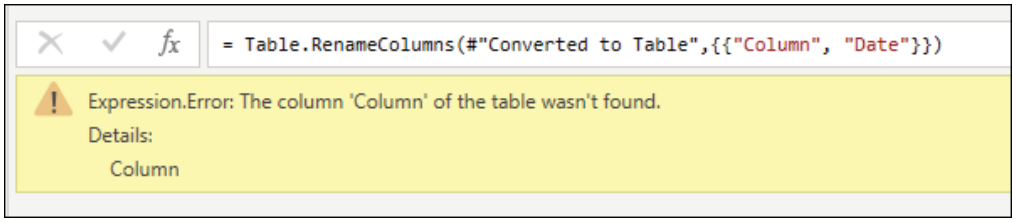


Figure 5.5.4.: Expression error, the column of the table wasn't found because the column name was manually changed.

Possible solutions: There are multiple solutions for this case, but they all depend on what you'd like to do. For this example, because the correct Date column header already comes from your text file, you can just remove the step that renames the column. This will allow your query to run without this error.

Other common step-level errors:

When combining or merging data between multiple data sources, you might get a Formula.Firewall error such as the one shown in the following image.

Figure 5.5.5.: Formula Firewall error, the query references other queries or steps, so it may not directly access a data source.

This error can be caused by a number of reasons, such as the data privacy levels between data sources or the way that these data sources are being combined or merged. For more information about how to diagnose this issue, go to Data privacy firewall.

5.5.2. CELL-LEVEL ERROR:

A cell-level error won't prevent the query from loading, but displays error values as Error in the cell. Selecting the white space in the cell displays the error pane underneath the data preview.

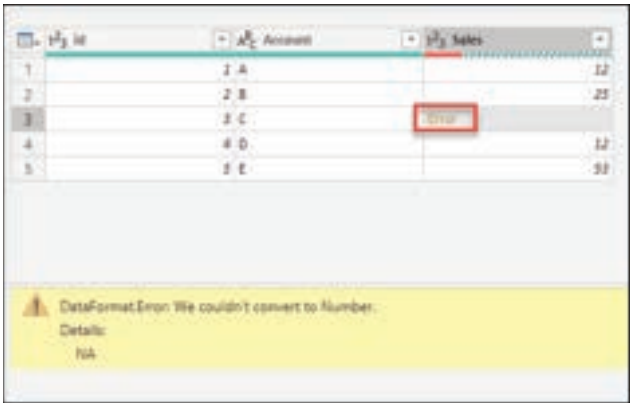


Figure 5.5.6: Cell level error

Handling errors at the cell level:

When encountering any cell-level errors, Power Query provides a set of functions to handle them either by removing, replacing, or keeping the errors.

For the next sections, the provided examples will be using the same sample query as the start point. In this query, you have a Sales column that has one cell with an error caused by a conversion error. The value inside that cell was NA, but when you transformed that column to a whole number Power Query couldn't convert NA to a number, so it displays the following error.

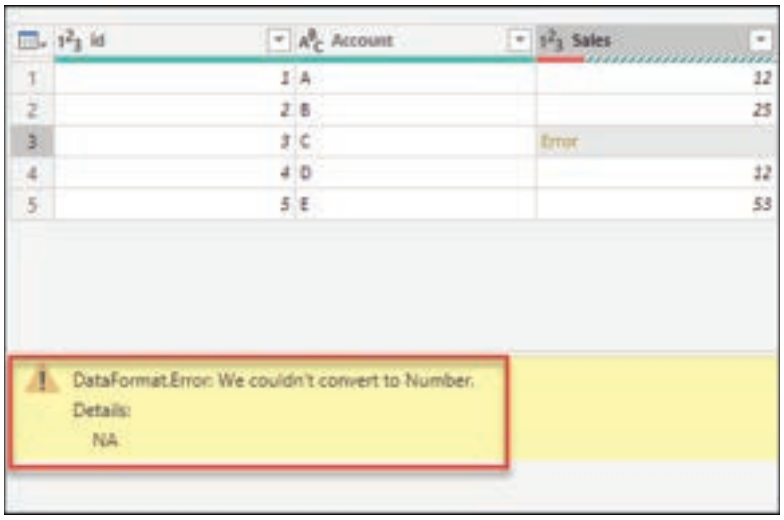


Figure 5.5.7.: Displays data format error, couldn't convert to data type error in the error pane.

Remove errors:

To remove rows with errors in Power Query, first select the column that contains errors. On the Home tab, in the Reduce rows group, select Remove rows. From the drop-down menu, select Remove errors.

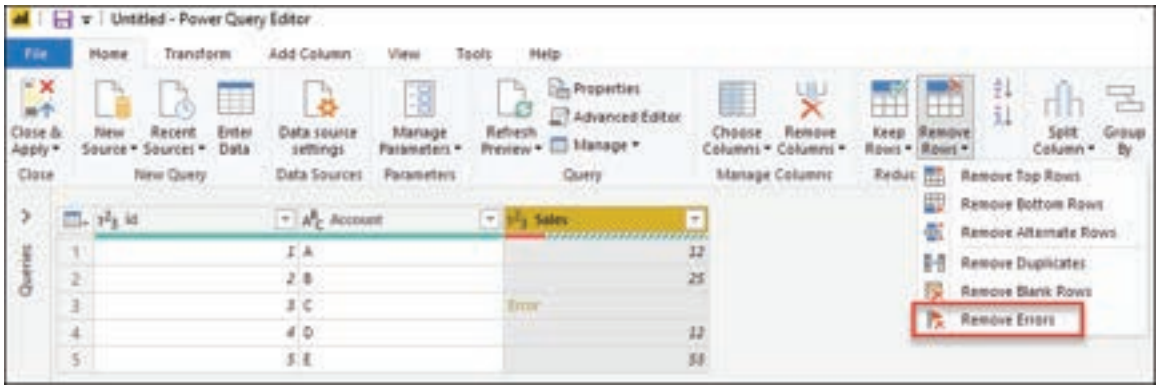


Figure 5.5.8.: Remove errors button on the Home tab.

The result of that operation will give you the table that you're looking for.

	id	Account	Sales
1	1	A	12
2	2	B	25
3	4	D	12
4	5	E	53

Figure 5.5.9.: Table that previously contained five rows now has the row that contained the error removed, leaving four rows in the table.

Replace errors:

If instead of removing rows with errors, you want to replace the errors with a fixed value, you can do so as well. To replace rows that have errors, first select the column that contains errors. On the Transform tab, in the Any column group, select Replace values. From the drop-down menu, select Replace errors.

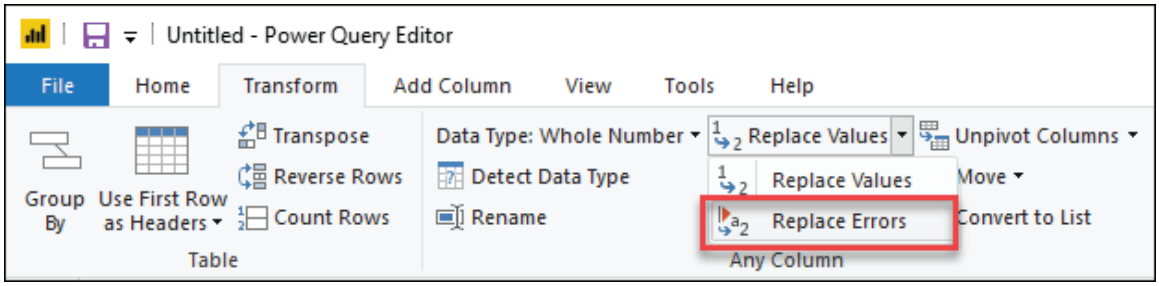


Figure 5.5.10.: Replace errors button on the Transform tab.

In the Replace errors dialog box, enter the value 10 because you want to replace all errors with the value 10.

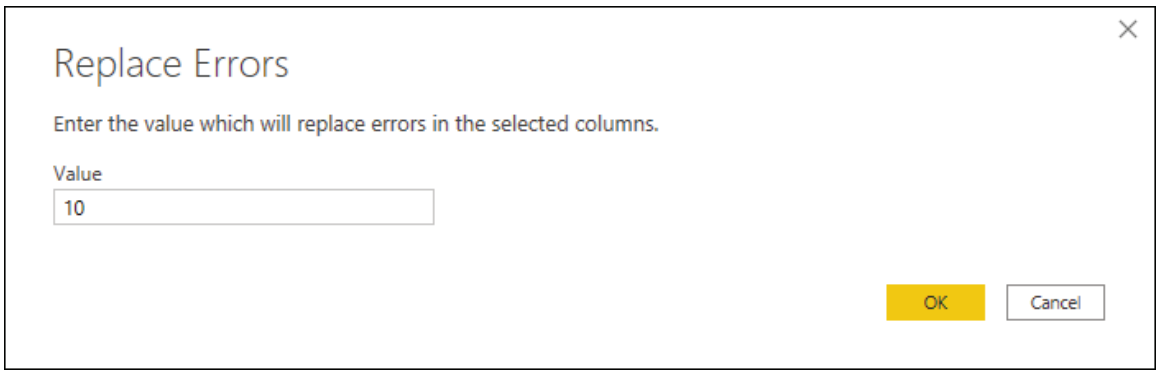


Figure 5.5.11.: Replace errors dialog box.

The result of that operation will give you the table that you're looking for.

	id	Account	Sales
1	1	A	12
2	2	B	25
3	3	C	10
4	4	D	12
5	5	E	53

Figure 5.5.12.: Table in which the third row contained an error in the Sales column now has the error replaced with the value 10.

Keep errors:

Power Query can serve as a good auditing tool to identify any rows with errors even if you don't fix the errors. This is where Keep errors can be helpful. To keep rows that have errors, first select the column that contains errors. On the Home tab, in the Reduce rows group, select Keep rows. From the drop-down menu, select Keep errors.

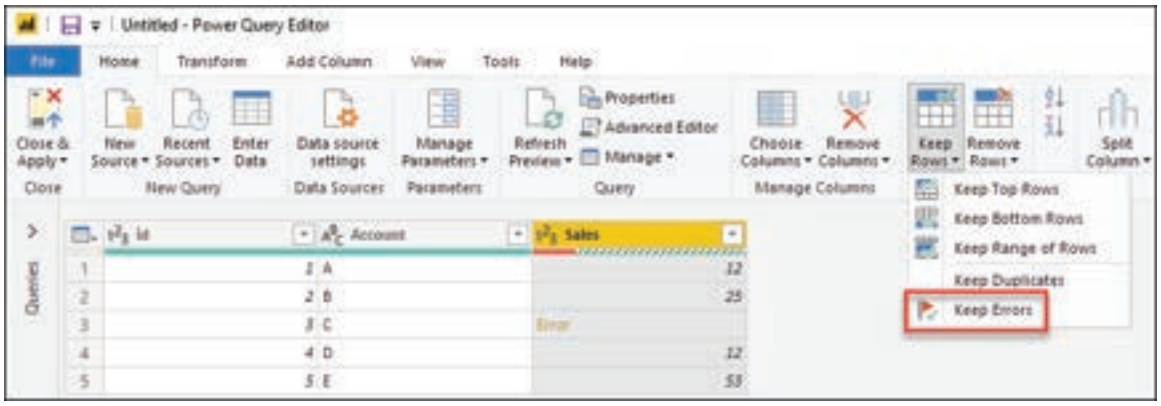


Figure 5.5.13.: Keep errors button on the Home tab.

The result of that operation will give you the table that you're looking for.

	id	Account	Sales
1	3	C	Error

Figure 5.5.14.: Final table that keeps only rows that contain errors.

Common cell-level errors:

As with any step-level error, we recommend that you take a close look at the error reasons, error messages, and error details provided at the cell level to understand what's causing the errors. The following sections discuss some of the most frequent cell-level errors in Power Query.

Data type conversion errors:

Commonly triggered when changing the data type of a column in a table. Some values found in the column could not be converted to the desired data type.

Example: You have a query that includes a column named Sales. One cell in that column has NA as a cell value, while the rest have whole numbers as values. You decide to convert the data type of the column from text to whole number, but the cell with the NA value causes an error.

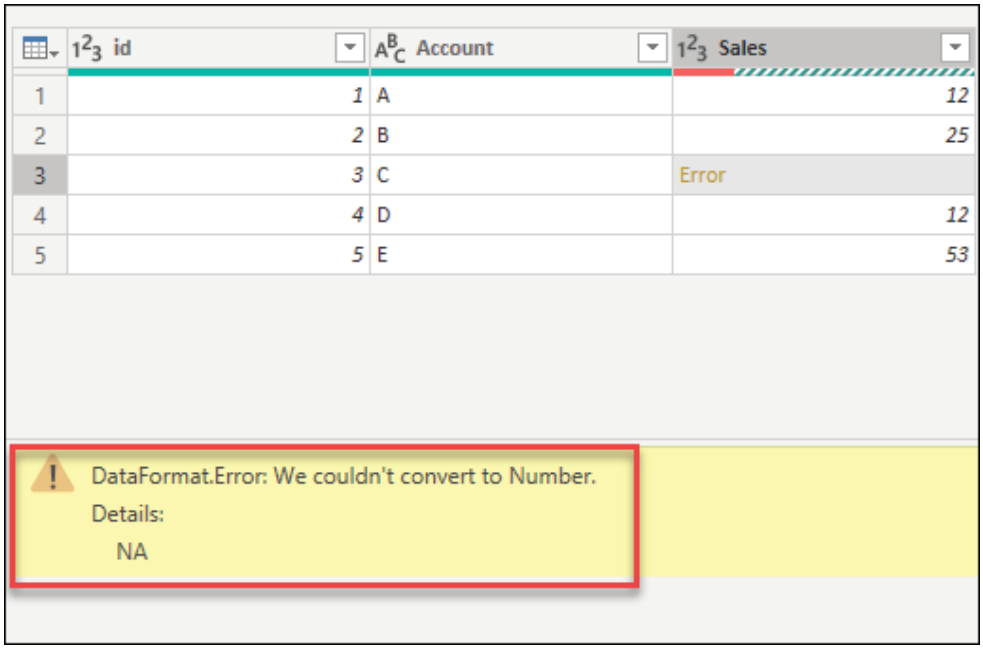


Figure 5.5.15.: Could not convert to data type error details.

Possible solutions: After identifying the row with the error, you can either modify the data source to reflect the correct value rather than NA, or you can apply a Replace error operation to provide a value for any NA values that cause an error.

Operation errors:

When trying to apply an operation that isn't supported, such as multiplying a text value by a numeric value, an error occurs.

Example: You want to create a custom column for your query by creating a text string that contains the phrase "Total Sales: " concatenated with the value from the Sales column. An error occurs because the concatenation operation only supports text columns and not numeric ones.

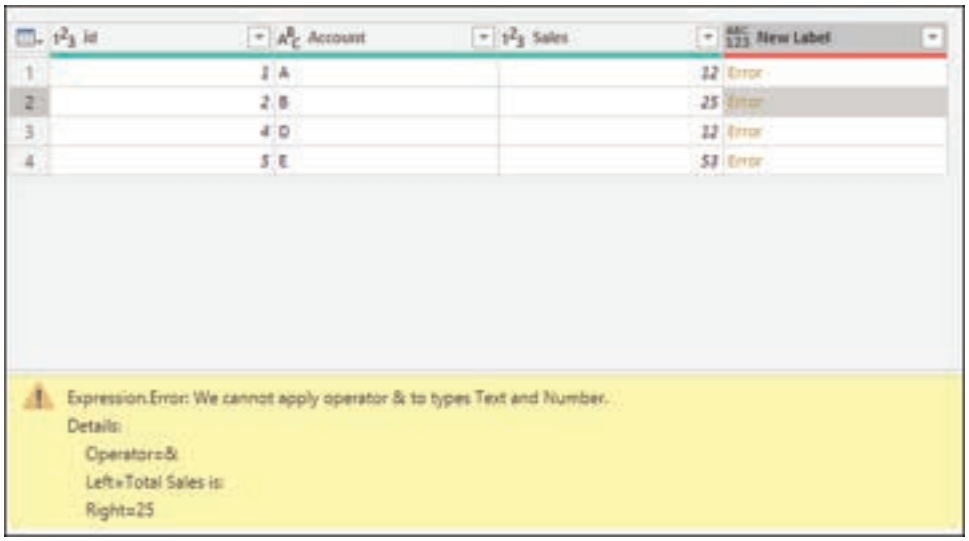


Figure 5.5.16.: Expression error in the error pane caused by trying to apply an And operator to text and a number from the Sales column.

Possible solutions: Before creating this custom column, change the data type of the Sales column to be text.

	id	Account	Sales	New Label
1	1	A	12	Total Sales is: 12
2	2	B	25	Total Sales is: 25
3	4	D	12	Total Sales is: 12
4	5	E	53	Total Sales is: 53

Figure 5.5.17.: Table with the sales column converted from a Number data type to a Text data type, and the resulting a new column containing both expressions.

Nested values shown as errors:

When working with data that contains nested structured values (such as tables, lists, or records), you may sometimes encounter the following error:

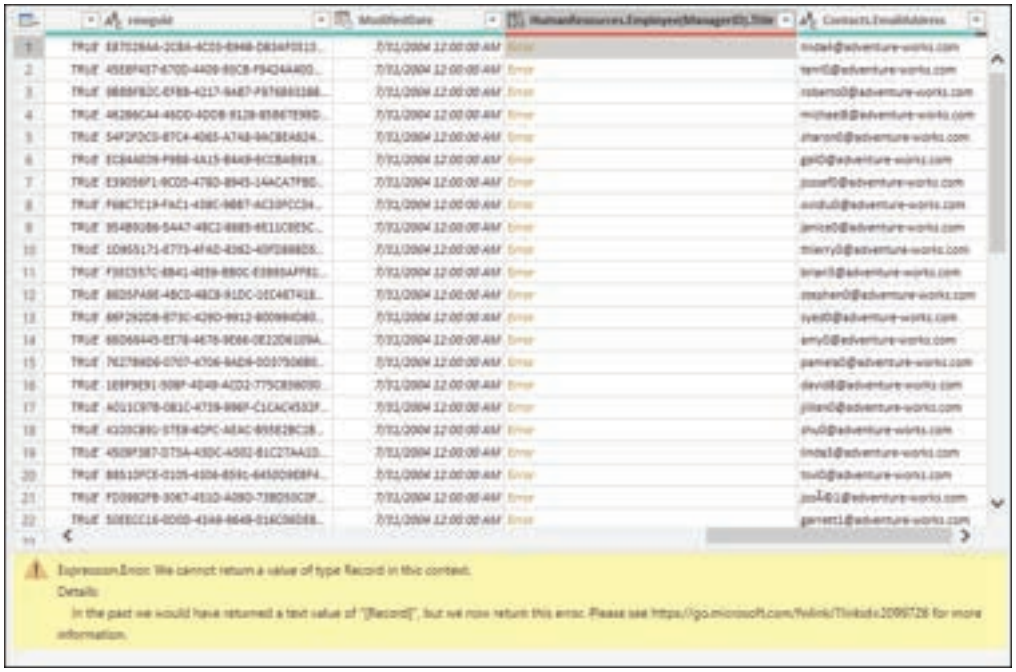


Figure 5.5.18.: Error for nested values triggered by formula firewall taken place

Expression.Error: We cannot return a value of type {value} in this context
Details: In the past we would have returned a text value of {value}, but we now return this error. Please see https://go.microsoft.com/fwlink/?linkid=2099726 for more information.

These errors usually occur for two reasons:

- When the Data Privacy Firewall buffers a data source, nested non-scalar values are automatically converted to errors.
- When a column defined with the Any data type contains non-scalar values, such values will be reported as errors during load (such as in a Workbook in Excel or the data model in Power BI Desktop).

Possible solutions:

- Remove the column that contains the error, or set a non-Any data type for such a column.
- Change the privacy levels of the data sources involved to one that allows them to be combined without being buffered.
- Flatten the tables before doing a merge to eliminate columns that contain nested structured values (such as table, record, or list).

SUMMARY

Step-level errors, which can halt the entire query, are explored in detail. Components of step-level errors, including error reason, message, and detail, are explained. Common step-level errors like "Can't find the source" and "The column of the table wasn't found" are highlighted, along with practical solutions for resolving them.

Cell-level errors, which do not prevent query execution but appear as error values, are covered extensively. Strategies for removing, replacing, or keeping cell-level errors are discussed. Common cell-level errors like data type conversion errors, operation errors, and nested value errors are identified, and solutions for addressing them are provided.

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 What are the two primary types of errors in Power Query?
   a) Syntax errors and runtime errors
   b) Step-level errors and cell-level errors
   c) Data connection errors and data transformation errors
   d) Input errors and output errors

2 Which of the following components make up a step-level error in Power Query?
   a) Error reason, error solution, error code
   b) Error type, error description, error summary
   c) Error reason, error message, error detail
   d) Error source, error impact, error resolution

3 In Power Query, what is the primary purpose of the "Go to error" button?
   a) It provides suggestions for fixing errors.
   b) It displays a detailed error report.
   c) It navigates to the first step where the error occurred.
   d) It removes errors from the query.

4 What is a common cause of the step-level error "Can't find the source" in Power Query?
   a) Incorrect data transformation functions
   b) Insufficient data source credentials
   c) Merging data from multiple sources
   d) Unsupported file formats

5 To address a step-level error related to a missing column, what should you consider doing?
   a) Ignore the error and proceed with the query.
   b) Rename the column in the query.
   c) Modify the data source to include the missing column.
   d) Delete the entire query and start over.

6 What is the main purpose of cell-level errors in Power Query?
   a) To stop the query from executing
   b) To display error values in cells
   c) To provide suggestions for fixing errors
   d) To improve query performance

7 When encountering cell-level errors in Power Query, what is one approach to handle them?
   a) Delete the entire column with errors.
   b) Replace errors with a fixed value.
   c) Ignore the errors and proceed with the query.
   d) Convert errors into step-level errors.

8 What does the "Keep errors" option in Power Query allow you to do?
   a) Automatically fix cell-level errors.
   b) Retain rows with cell-level errors.
   c) Delete all rows with errors.
   d) Convert cell-level errors into step-level errors.

9 Which of the following is a common cell-level error in Power Query?
   a) Step not found error
   b) Data source inaccessible error
   c) Data type conversion error
   d) Formula.Firewall error

10 What is a possible solution for a data type conversion error in Power Query?
   a) Modify the data source to include the missing column.
   b) Delete the entire query.
   c) Convert the column to a different data type.
   d) Ignore the error and proceed with the query.

11 When might you encounter an operation error in Power Query?
   a) When combining data from multiple sources
   b) When renaming a column
   c) When applying an unsupported operation, like multiplying text by a number
   d) When sorting data in ascending order

12 What is the primary cause of nested value errors in Power Query?
   a) Incomplete data sources
   b) Mismatched data privacy levels
   c) Incorrect data transformations
   d) Formula.Firewall errors

13 Which of the following solutions is recommended for resolving nested value errors in Power Query?
   a) Remove the column that contains the error.
   b) Flatten the tables before merging.
   c) Convert all columns to the Any data type.
   d) Ignore the error and continue with the query.

14 What is the primary goal when handling step-level errors in Power Query?
   a) Remove all errors to ensure query execution.
   b) Understand the error reasons, messages, and details to fix issues.
   c) Replace errors with the "Error" value to maintain data integrity.
   d) Convert step-level errors into cell-level errors.

15 In Power Query, what is the recommended action when encountering a step-level error related to data source access?
   a) Delete the entire query.
   b) Modify the error message.
   c) Check data source credentials and accessibility.
   d) Convert the error into a cell-level error.

Answers

1 *B) Step-level errors and cell-level errors*
2 *C) Error reason, error message, error detail*
3 *C) It navigates to the first step where the error occurred.*
4 *B) Insufficient data source credentials*
5 *C) Modify the data source to include the missing column.*
6 *B) To display error values in cells*
7 *B) Replace errors with a fixed value.*
8 *B) Retain rows with cell-level errors.*
9 *C) Data type conversion error*
10 *C) Convert the column to a different data type.*
11 *C) When applying an unsupported operation, like multiplying text by a number*
12 *B) Mismatched data privacy levels*
13 *B) Flatten the tables before merging.*
14 *B) Understand the error reasons, messages, and details to fix issues.*
15 *C) Check data source credentials and accessibility.*

SELF-EXAMINATION QUESTIONS FOR PRACTICE:

1 What are the two main types of errors discussed in Power Query, and how do they differ in their impact on query execution?

2 Can you explain the components of a step-level error in Power Query, including error reason, error message, and error detail?

3 What is the significance of the "Go to error" button in Power Query, and when is it recommended to use it?

4 Provide an example and possible solutions for the common step-level error "Can't find the source" in Power Query.

5 When encountering a step-level error related to a missing column, how can you resolve it, and what considerations should be made?

6 Describe the purpose of cell-level errors in Power Query and how they are displayed in the interface.

7 Explain the three approaches to handling cell-level errors in Power Query: removing, replacing, and keeping errors. When might each approach be most useful?

8 Give examples of common cell-level errors, such as data type conversion errors and operation errors, and provide guidance on how to address them effectively.

CHAPTER 5

5.6. DATA PROFILING:

LEARNING OBJECTIVES

Upon completing this content, you will be able to effectively utilize the data profiling tools in Power Query Editor to clean, transform, and understand data. Specifically, will be able to:

- Enable the data profiling tools in both Power Query Desktop and Power Query Online.
- Interpret and utilize the Column Quality feature to identify and manage data quality issues, including recognizing valid, error, empty, unknown, and unexpected error values within columns.
- Navigate and interpret the visual representations of data distributions provided by the Column Distribution tool, understanding how to analyze and perform operations on the data.
- Utilize the Column Profile feature to gain a deeper understanding of column data, including statistics and value distributions, and employ actions such as filtering, copying, and grouping data for further analysis and transformation.

The data profiling tools provide new and intuitive ways to clean, transform, and understand data in Power Query Editor. They include:

- Column quality
- Column distribution
- Column profile

To enable the data profiling tools, go to the View tab on the ribbon. In Power Query Desktop, enable the options you want in the Data preview group, as shown in the following image.

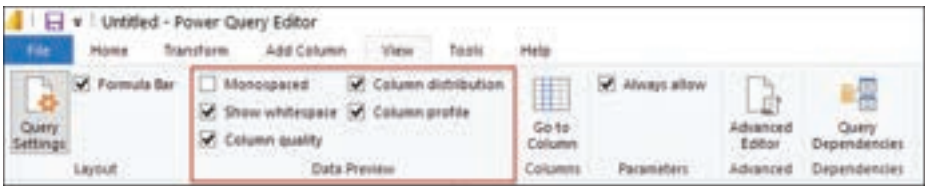


Figure 5.6.1.: Data profiling tools.

In Power Query Online, select Data view, then enable the options you want in the drop-down list

Figure 5.6.2.: Query settings

After you enable the options, you'll see something like the following image in Power Query Editor.

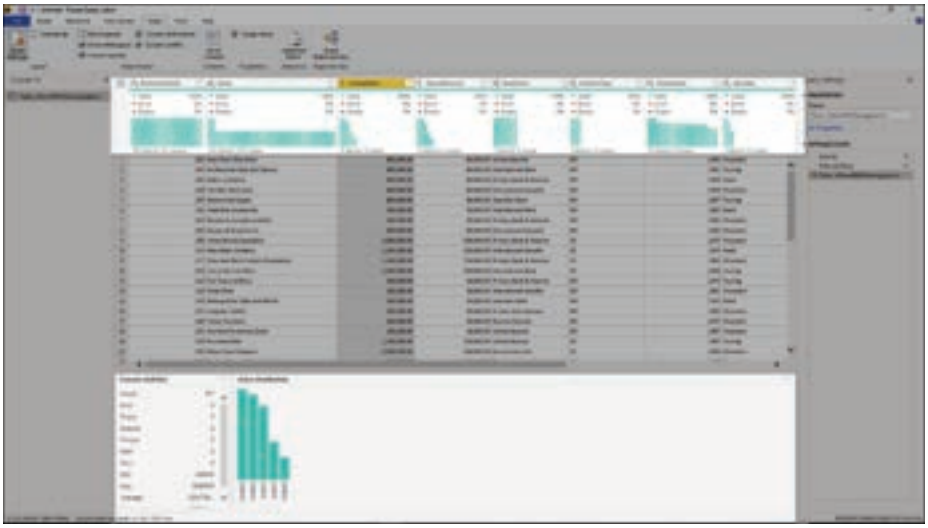


Figure 5.6.3.: Data profiling tools enabled.

5.6.1. COLUMN QUALITY:

The column quality feature labels values in rows in five categories:

- Valid, shown in green.
- Error, shown in red.
- Empty, shown in dark grey.
- Unknown, shown in dashed green. Indicates when there are errors in a column, the quality of the remaining data is unknown.
- Unexpected error, shown in dashed red.

These indicators are displayed directly underneath the name of the column as part of a small bar chart, as shown in the following image.

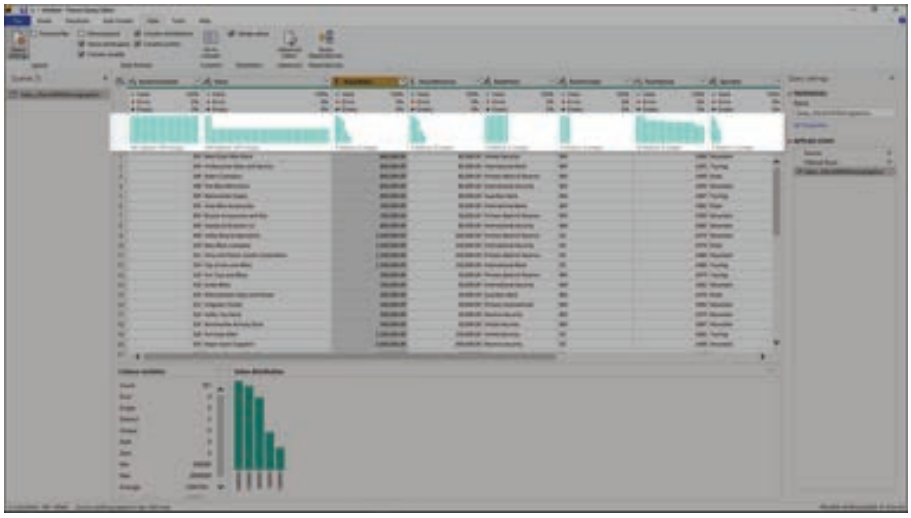


Figure 5.6.4.: Enhanced view of the bar chart with data quality indicators and labels above each column in the table.

The number of records in each column quality category is also displayed as a percentage.

By hovering over any of the columns, you are presented with the numerical distribution of the quality of values throughout the column. Additionally, selecting the ellipsis button (...) opens some quick action buttons for operations on the values.

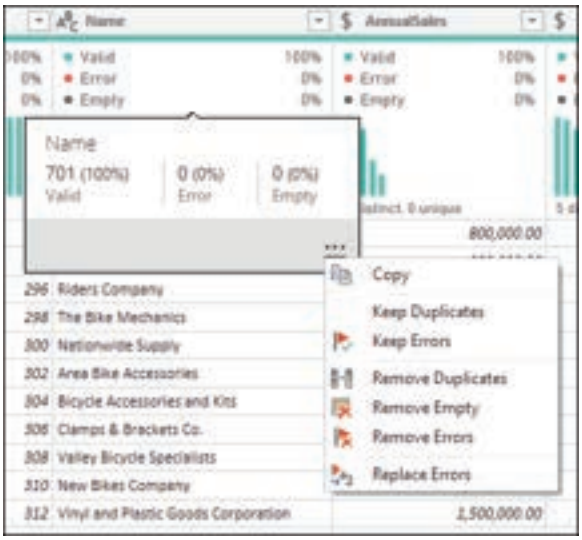


Figure 5.6.5.: Distribution of names column with 701 names valid (100 percent), zero errors and zero empty, with quick action commands displayed.

5.6.2. COLUMN DISTRIBUTION:

This feature provides a set of visuals underneath the names of the columns that showcase the frequency and distribution of the values in each of the columns. The data in these visualizations is sorted in descending order from the value with the highest frequency.

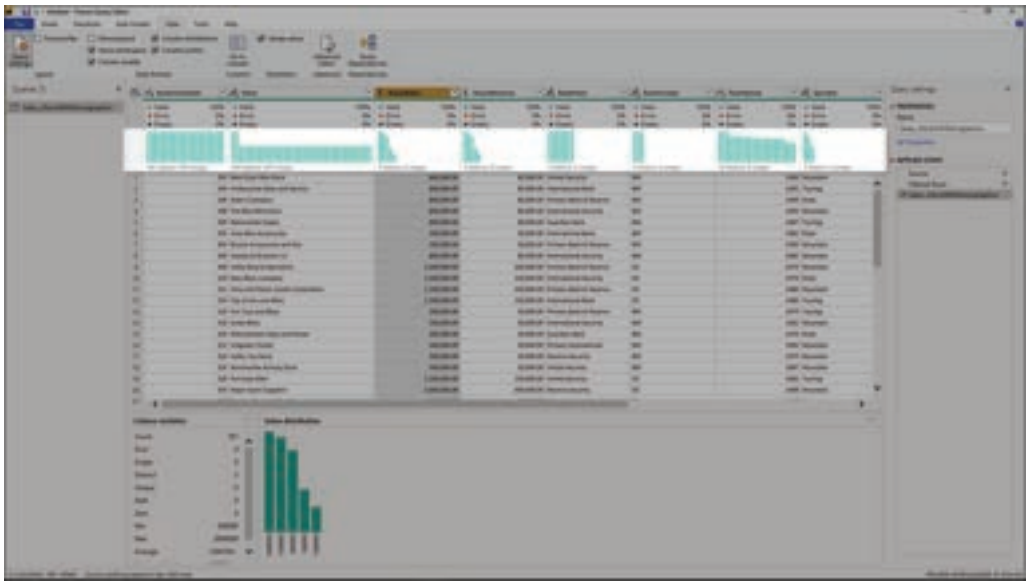


Figure 5.6.6.: Column distribution.

By hovering over the distribution data in any of the columns, you get information about the overall data in the column (with distinct count and unique values). You can also select the ellipsis button and choose from a menu of available operations.

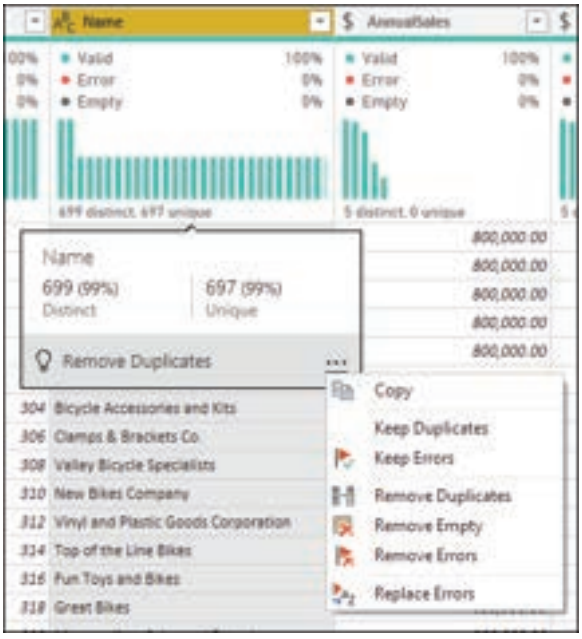


Figure 5.6.7.: Column distributions options.

5.6.3. COLUMN PROFILE:

This feature provides a more in-depth look at the data in a column. Apart from the column distribution chart, it contains a column statistics chart. This information is displayed underneath the data preview section, as shown in the following image.

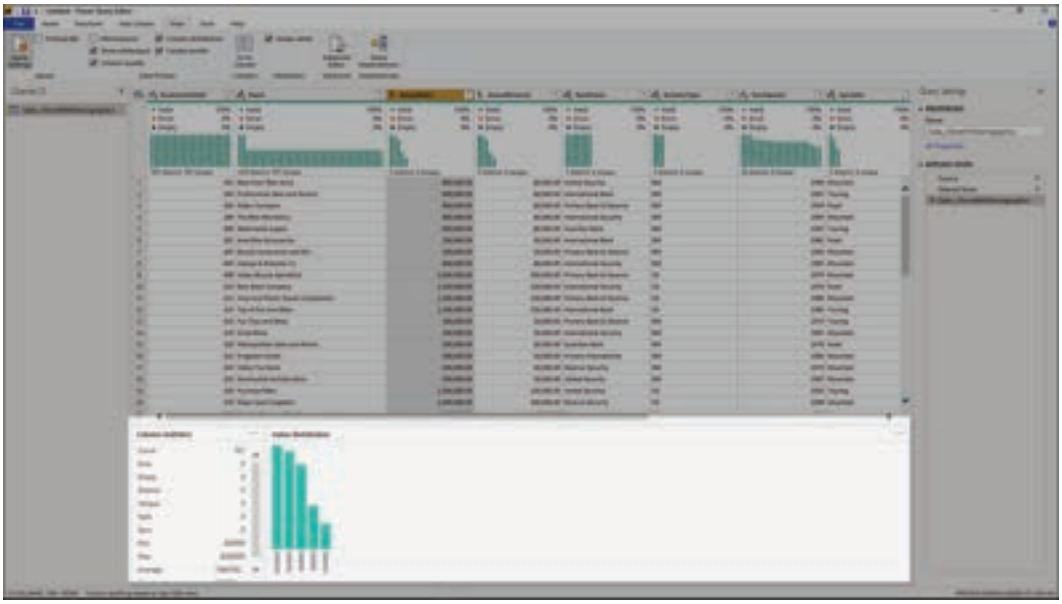


Figure 5.6.8.: Column profile.

Filter by value:

You can interact with the value distribution chart on the right side and select any of the bars by hovering over the parts of the chart.

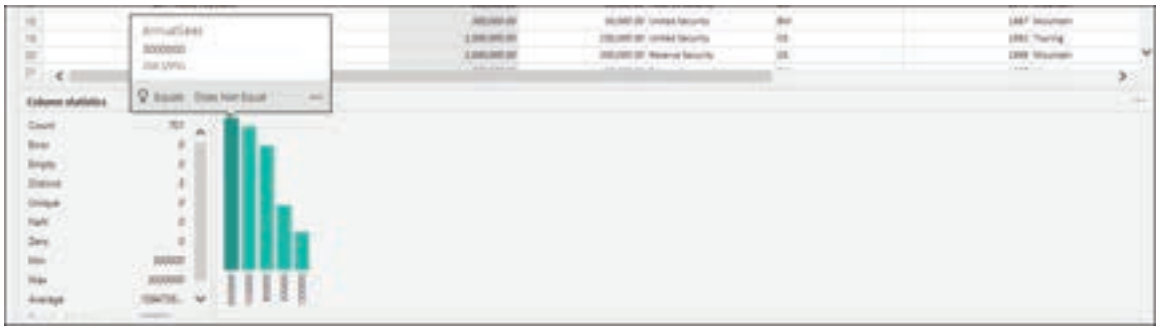


Figure 5.6.9.: Hover over bar in value distribution bar chart, which shows individual data for that bar.

Right-click to display a set of available transformations for that value.

Figure 5.6.10.: Displays shortcut menu with available transformations for a single bar in the value distribution bar chart.

Copy data:

In the upper-right corner of both the column statistics and value distribution sections, you can select the ellipsis button (...) to display a Copy shortcut menu. Select it to copy the data displayed in either section to the clipboard.

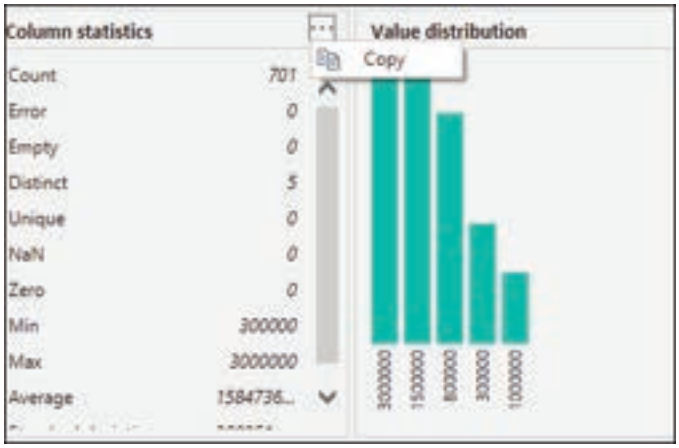


Figure 5.6.11.: Copy column statistics.

Group by value:

When you select the ellipsis button (...) in the upper-right corner of the value distribution chart, in addition to Copy you can select Group by. This feature groups the values in your chart by a set of available options.

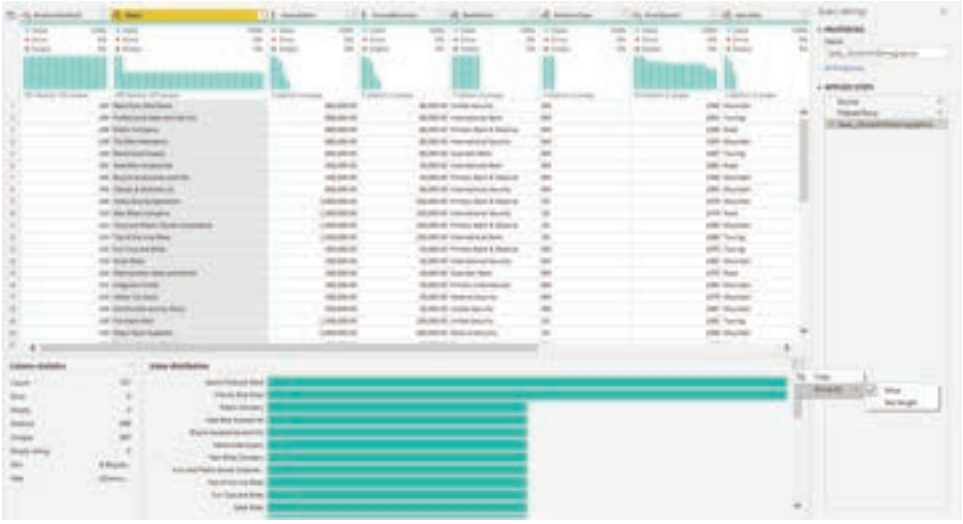


Figure 5.6.12.: Group by value distribution.

The image below shows a column of product names that have been grouped by text length. After the values have been grouped in the chart, you can interact with individual values in the chart as described in Filter by value.

Figure 5.6.13.: New bar chart showing the distribution of the text length of store names in the table's Names column.

SUMMARY

- Enabling Data Profiling Tools: Learners will learn how to activate the data profiling tools in both Power Query Desktop and Power Query Online through the View tab, gaining access to powerful data analysis capabilities.

- Column Quality Analysis: Understand how to utilize the Column Quality feature to categorize data into five categories (Valid, Error, Empty, Unknown, Unexpected Error). Interpret visual cues and percentages for effective data quality assessment and troubleshooting.

- Column Distribution Insights: Explore the Column Distribution tool, which provides visuals illustrating the frequency and distribution of column values. Master the ability to analyze data characteristics such as distinct counts and unique values while applying various data analysis operations.

- Column Profiling: Attain a deep grasp of column data through the Column Profile feature, which includes statistics and value distribution charts. Learn how to filter, copy, and group data, enabling further in-depth data analysis and transformation

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 How can users enable data profiling tools in Power Query Editor?
   a) Through the Data tab
   b) Through the Edit tab
   c) Through the View tab
   d) Through the Home tab

2 What are the five categories used by the Column Quality feature to label data values?
   a) Good, Bad, Empty, Unknown, Undecided
   b) Valid, Invalid, Null, Unknown, Ambiguous
   c) Valid, Error, Empty, Unknown, Unexpected Error
   d) Correct, Incorrect, Missing, Unidentified, Abnormal

3 What is the main purpose of visual indicators and percentages when using the Column Quality feature?
   a) To add visual appeal to the data
   b) To calculate statistical measures
   c) To identify and address data quality issues
   d) To create data charts

4 Which tool provides visuals depicting the frequency and distribution of column values?
   a) Column Quality
   b) Column Profiling
   c) Data Distribution
   d) Column Distribution

5 What type of data characteristics can users explore when working with the Column Distribution tool?
   a) Unique identifiers
   b) Data types
   c) Distinct counts and unique values
   d) Data format errors

6 What does the Column Profile feature offer in addition to value distribution charts?
   a) Data categorization
   b) Statistical insights and value distribution
   c) Data filtering options
   d) Data transformation capabilities

7 How can users filter data when using the Column Profile feature?
   a) By sorting values in ascending order
   b) By selecting distinct values
   c) By using the ellipsis button (...)
   d) By interacting with the value distribution chart

8 Which option allows users to copy the data displayed in either the column statistics or value distribution sections?
   a) Right-clicking
   b) Selecting the Filter option
   c) Using the Group by feature
   d) Selecting the ellipsis button (...)

9 What additional feature is available when users select the ellipsis button (...) in the value distribution chart besides copying data?
   a) Renaming the column
   b) Grouping values
   c) Deleting the column
   d) Group by

10 Why might a user want to group data by value using the Group by feature in the Column Profile tool?
   a) To create new columns
   b) To merge columns
   c) To perform advanced calculations
   d) To analyze data based on specific characteristics

11 In Power Query Editor, where can users find the option to enable data profiling tools?
   a) Data tab
   b) Edit tab
   c) View tab
   d) Transform tab

12 Which category of data values is indicated by dashed red lines in the Column Quality feature?
   a) Valid
   b) Error
   c) Empty
   d) Unknown

13 What information does the Column Distribution tool display in descending order?
   a) Values with the highest quality
   b) Values with the lowest frequency
   c) Values with the highest frequency
   d) Values with the lowest quality

14 What action can users perform when they hover over the value distribution chart in the Column Profile tool?
   a) Delete values
   b) Add new values
   c) Get information about the data in the column
   d) Create pivot tables

15 What does the ellipsis button (...) in the Column Profile tool allow users to do besides copying data?
   a) Export data to Excel
   b) Perform advanced statistical analysis
   c) Group data by value
   d) Share data on social media

Answers

1 *C) Through the View tab*
2 *C) Valid, Error, Empty, Unknown, Unexpected Error*
3 *C) To identify and address data quality issues*
4 *D) Column Distribution*
5 *C) Distinct counts and unique values*
6 *B) Statistical insights and value distribution*
7 *D) By interacting with the value distribution chart*
8 *D) Selecting the ellipsis button (...)*
9 *D) Group by*
10 *D) To analyze data based on specific characteristics*
11 *C) View tab*
12 *B) Error*
13 *C) Values with the highest frequency*
14 *C) Get information about the data in the column*
15 *C) Group data by value*

SELF-EXAMINATION QUESTIONS FOR PRACTICE:

1 How can users enable the data profiling tools in Power Query Desktop and Power Query Online?

2 What is the purpose of the Column Quality feature, and how does it categorize data values?

3 Can you explain the five categories used by the Column Quality feature to label data values?

4 What role do visual indicators and percentages play in data quality analysis using the Column Quality feature?

5 How does the Column Distribution tool assist users in understanding the distribution of column values?

6 What are some key insights that can be obtained by exploring the data characteristics provided by the Column Distribution tool?

7 When using the Column Profile feature, what types of information can users access about their data?

8 How can users filter data using the Column Profile feature, and why is this capability valuable?

9 What options are available for copying data when working with the data profiling tools?

10 Describe the significance of grouping data by value using the Column Profile feature, and provide an example of when this might be useful in data analysis.

CHAPTER 5

5.7. POWER QUERY FORMULA LANGUAGE:

LEARNING OBJECTIVES

Upon completing this content, you will be able to understand and work with the Microsoft Power Query formula language (M) by:

- Describing the lexical structure of M.
- Recognizing and using values, expressions, environments, variables, and identifiers in M.
- Differentiating between primitive and structured values in M.
- Explaining the concept of types in M.
- Utilizing operators in M expressions.
- Defining and applying functions in M.
- Handling errors in M expressions.
- Utilizing Let expressions for defining auxiliary values.
- Implementing If expressions for conditional evaluation.
- Understanding the concept of sections in M.
- Comprehending the overall grammar and structure of M.

Microsoft Power Query provides a powerful "get data" experience that encompasses many features. A core capability of Power Query is to filter and combine, that is, to "mash-up" data from one or more of a rich collection of supported data sources. Any such data mashup is expressed using the Power Query formula language (informally known as "M"). Power Query embeds M documents in a wide range of Microsoft products, including Excel, Power BI, Analysis Services, and Dataverse, to enable repeatable mashup of data.

This document provides the specification for M. After a brief introduction that aims at building some first intuition and familiarity with the language, the document covers the language precisely in several progressive steps:

- The lexical structure defines the set of texts that are lexically valid.
- Values, expressions, environments and variables, identifiers, and the evaluation model form the language's basic concepts.
- The detailed specification of values, both primitive and structured, defines the target domain of the language.
- Values have types, themselves a special kind of value, that both characterize the fundamental kinds of values and carry additional metadata that is specific to the shapes of structured values.
- The set of operators in M defines what kinds of expressions can be formed.
- Functions, another kind of special values, provide the foundation for a rich standard library for M and allow for the addition of new abstractions.

- Errors can occur when applying operators or functions during expression evaluation. While errors aren't values, there are ways to handle errors that map errors back to values.
- Let expressions allow for the introduction of auxiliary definitions used to build up complex expressions in smaller steps.
- If expressions support conditional evaluation.
- Sections provide a simple modularity mechanism. (Sections aren't yet leveraged by Power Query.)
- Finally, a consolidated grammar collects the grammar fragments from all other sections of this document into a single complete definition.

For computer language theorists: the formula language specified in this document is a mostly pure, higher-order, dynamically typed, partially lazy functional language.

5.7.1 EXPRESSIONS AND VALUES: M

The central construct in M is the expression. An expression can be evaluated (computed), yielding a single value.

Although many values can be written literally as an expression, a value isn't an expression. For example, the expression 1 evaluates to the value 1; the expressions 1+1 evaluates to the value 2. This distinction is subtle, but important. Expressions are recipes for evaluation; values are the results of evaluation.

The following examples illustrate the different kinds of values available in M. As a convention, a value is written using the literal form in which they would appear in an expression that evaluates to just that value. (Note that the // indicates the start of a comment which continues to the end of the line.)

- A primitive value is single-part value, such as a number, logical, text, or null. A null value can be used to indicate the absence of any data.

```
Power Query M

123                 // A number
true                // A logical
"abc"               // A text
null                // null value
```

- A list value is an ordered sequence of values. M supports infinite lists, but if written as a literal, lists have a fixed length. The curly brace characters { and } denote the beginning and end of a list.

```
Power Query M

{123, true, "A"}     // list containing a number, a logical, and
                     //     a text
{1, 2, 3}            // list of three numbers
```

- A record is a set of fields. A field is a name/value pair where the name is a text value that's unique within the field's record. The literal syntax for record values allows the names to be written without quotes, a form also referred to as identifiers. The following shows a record containing three fields named "A", "B", and "C", which have values 1, 2, and 3.

```
Power Query M

[
      A = 1,
      B = 2,
      C = 3
]
```

- A table is a set of values organized into columns (which are identified by name), and rows. There's no literal syntax for creating a table, but there are several standard functions that can be used to create tables from lists or records.

```
Power Query M

#table( {"A", "B"}, { {1, 2}, {3, 4} } )
```

This creates a table of the following shape:

A	B
1	2
3	4

Figure 5.7.1 Image of an example table in the M formula language.

A function is a value that, when invoked with arguments, produces a new value. A function is written by listing the function's parameters in parentheses, followed by the goes-to symbol =>, followed by the expression defining the function. That expression typically refers to the parameters (by name).

```
Power Query M

(x, y) => (x + y) / 2`
```

5.7.2. EVALUATION:

The evaluation model of the M language is modeled after the evaluation model commonly found in spreadsheets, where the order of calculation can be determined based on dependencies between the formulas in the cells.

If you've written formulas in a spreadsheet such as Excel, you may recognize the formulas on the left result in the values on the right when calculated:

	A
1	=A2 * 2
2	=A3 + 1
3	1

	A
1	4
2	2
3	1

Figure 5.7.2.: Formulas on the right resulting in the values on the left.

In M, parts of an expression can reference other parts of the expression by name, and the evaluation process automatically determines the order in which referenced expressions are calculated.

You can use a record to produce an expression that's equivalent to the previous spreadsheet example. When initializing the value of a field, you can refer to other fields within the record by using the name of the field, as follows:

```
Power Query M

[
    A1 = A2 * 2,
    A2 = A3 + 1,
    A3 = 1
]
```

The above expression is equivalent to the following (in that both evaluate to equal values):

```
Power Query M

[
    A1 = 4,
    A2 = 2,
    A3 = 1
]
```

Records can be contained within, or nest, within other records. You can use the lookup operator ([]) to access the fields of a record by name. For example, the following record has a field named Sales containing a record, and a field named Total that accesses the FirstHalf and SecondHalf fields of the Sales record:

```
Power Query M

[
    Sales = [ FirstHalf = 1000, SecondHalf = 1100 ],
    Total = Sales[FirstHalf] + Sales[SecondHalf]
]
```

The above expression is equivalent to the following when it is evaluated:

```
Power Query M

[
    Sales = [ FirstHalf = 1000, SecondHalf = 1100 ],
    Total = 2100
]
```

Records can also be contained within lists. You can use the positional index operator ({}) to access an item in a list by its numeric index. The values within a list are referred to using a zero-based index from the beginning of the list. For example, the indexes 0 and 1 are used to reference the first and second items in the list below:

```
Power Query M

[
    Sales =
        {
            [
                Year = 2007,
                FirstHalf = 1000,
                SecondHalf = 1100,
                Total = FirstHalf + SecondHalf // 2100
            ],
            [
                Year = 2008,
                FirstHalf = 1200,
                SecondHalf = 1300,
                Total = FirstHalf + SecondHalf // 2500
            ]
        },
    TotalSales = Sales{0}[Total] + Sales{1}[Total] // 4600
]
```

List and record member expressions (as well as let expressions) are evaluated using lazy evaluation, which means that they are evaluated only as needed. All other expressions are evaluated using eager evaluation, which means that they are evaluated immediately when encountered during the evaluation process. A good way to think about this is to remember that evaluating a list or record expression returns a list or record value that itself remembers how its list items or record fields need to be computed, when requested (by lookup or index operators).

5.7.3. FUNCTIONS:

In M, a function is a mapping from a set of input values to a single output value. A function is written by first naming the required set of input values (the parameters to the function) and then providing an expression that computes the result of the function using those input values (the body of the function) following the goes-to (=>) symbol. For example:

```
Power Query M

(x) => x + 1                    // function that adds one to a value
(x, y) =>  x + y                // function that adds two values
```

List and record member expressions (as well as let expressions) are evaluated using lazy evaluation, which means that they are evaluated only as needed. All other expressions are evaluated using eager evaluation, which means that they are evaluated immediately when encountered during the evaluation process. A good way to think about this is to remember that evaluating a list or record expression returns a list or record value that itself remembers how its list items or record fields need to be computed, when requested (by lookup or index operators).

```
Power Query M

[
    Add = (x, y) => x + y,
    OnePlusOne = Add(1, 1),     // 2
    OnePlusTwo = Add(1, 2)      // 3
]
```

5.7.4. LIBRARY

M includes a common set of definitions available for use from an expression called the standard library, or just library for short. These definitions consist of a set of named values. The names of values provided by a library are available for use within an expression without having been defined explicitly by the expression. For example:

```
Power Query M

Number.E                        // Euler's number e (2.7182...)
Text.PositionOf("Hello", "ll")  // 2
```

5.7.5. OPERATORS

M includes a set of operators that can be used in expressions. Operators are applied to operands to form symbolic expressions. For example, in the expression 1 + 2 the numbers 1 and 2 are operands and the operator is the addition operator (+).

The meaning of an operator can vary depending on what kind of values its operands are. For example, the plus operator can be used with other kinds of values besides numbers:

```
Power Query M

1 + 2                    // numeric addition: 3
#time(12,23,0) + #duration(0,0,2,0)
                         // time arithmetic: #time(12,25,0)
```

Another example of an operator with operand-depending meaning is the combination operator (&):

```
Power Query M

"A" & "BC"              // text concatenation: "ABC"
{1} & {2, 3}            // list concatenation: {1, 2, 3}
[ a = 1 ] & [ b = 2 ]   // record merge: [ a = 1, b = 2 ]
```

Note that some operators don't support all combinations of values. For example:

```
Power Query M

1 + "2"  // error: adding number and text isn't supported
```

Expressions that, when evaluated, encounter undefined operator conditions evaluate to errors.

5.7.6. METADATA:

Metadata is information about a value that's associated with a value. Metadata is represented as a record value, called a metadata record. The fields of a metadata record can be used to store the metadata for a value.

Every value has a metadata record. If the value of the metadata record hasn't been specified, then the metadata record is empty (has no fields).

Metadata records provide a way to associate additional information with any kind of value in an unobtrusive way. Associating a metadata record with a value doesn't change the value or its behavior.

A metadata record value y is associated with an existing value x using the syntax x meta y. For example, the following code associates a metadata record with Rating and Tags fields with the text value "Mozart":

```
Power Query M

"Mozart" meta [ Rating = 5, Tags = {"Classical"} ]
```

For values that already carry a non-empty metadata record, the result of applying meta is that of computing the record merge of the existing and the new metadata record. For example, the following two expressions are equivalent to each other and to the previous expression:

```
Power Query M

("Mozart" meta [ Rating = 5 ]) meta [ Tags = {"Classical"} ]
"Mozart" meta ([ Rating = 5 ] & [ Tags = {"Classical"} ])
```

A metadata record can be accessed for a given value using the Value.Metadata function. In the following example, the expression in the ComposerRating field accesses the metadata record of the value in the Composer field, and then accesses the Rating field of the metadata record.

```
Power Query M

[
    Composer = "Mozart" meta [ Rating = 5, Tags = {"Classical"} ],
    ComposerRating = Value.Metadata(Composer)[Rating] // 5
]
```

5.7.7. LET EXPRESSION:

Many of the examples shown so far have included all the literal values of the expression in the result of the expression. The let expression allows a set of values to be computed, assigned names, and then used in a subsequent expression that follows the in. For example, in our sales data example, you could do:

```
Power Query M

let
    Sales2007 =
        [
            Year = 2007,
            FirstHalf = 1000,
            SecondHalf = 1100,
            Total = FirstHalf + SecondHalf // 2100
        ],
    Sales2008 =
        [
            Year = 2008,
            FirstHalf = 1200,
            SecondHalf = 1300,
            Total = FirstHalf + SecondHalf // 2500
        ]
  in Sales2007[Total] + Sales2008[Total] // 4600
```

The result of the above expression is a number value (4600) that's computed from the values bound to the names Sales2007 and Sales2008.

5.7.8. IF EXPRESSION:

The if expression selects between two expressions based on a logical condition. For example:

```
Power Query M

if 2 > 1 then
    2 + 2
else
    1 + 1
```

The first expression (2 + 2) is selected if the logical expression (2 > 1) is true, and the second expression (1 + 1) is selected if it's false. The selected expression (in this case 2 + 2) is evaluated and becomes the result of the if expression (4).

5.7.9. ERRORS:

An error is an indication that the process of evaluating an expression couldn't produce a value. Errors are raised by operators and functions encountering error conditions or by using the error expression. Errors are handled using the try expression. When an error is raised, a value is specified that can be used to indicate why the error occurred.

```
Power Query M

let Sales =
    [
        Revenue = 2000,
        Units = 1000,
        UnitPrice = if Units = 0 then error "No Units"
                    else Revenue / Units
    ],
    UnitPrice = try Number.ToText(Sales[UnitPrice])
in "Unit Price: " &
    (if UnitPrice[HasError] then UnitPrice[Error][Message]
    else UnitPrice[Value])
```

The above example accesses the Sales [Unit Price] field and formats the value producing the result:

Power Query M

```
"Unit Price: 2"
```

If the Units field had been zero, then the Unit Price field would have raised an error, which would have been handled by the try. The resulting value would then have been:

Power Query M

```
"No Units"
```

A try expression converts proper values and errors into a record value that indicates whether the try expression handled an error, or not, and either the proper value or the error record it extracted when handling the error. For example, consider the following expression that raises an error and then handles it right away:

Power Query M

```
try error "negative unit count"
```

This expression evaluates to the following nested record value, explaining the [HasError], [Error], and [Message] field lookups in the previous unit-price example.

Power Query M

```
[
    HasError = true,
    Error =
        [
            Reason = "Expression.Error",
            Message = "negative unit count",
            Detail = null
        ]
]
```

A common case is to replace errors with default values. The try expression can be used with an optional otherwise clause to achieve just that in a compact form:

```
Power Query M

try error "negative unit count" otherwise 42
// 42
```

SUMMARY

Powerful Data Retrieval: Microsoft Power Query offers a robust "get data" experience with extensive features.

Filter and Combine: A core feature of Power Query is the ability to filter and combine data from various supported sources, allowing users to create data mashups.

M Language Integration: Power Query incorporates the M formula language into multiple Microsoft products, including Excel, Power BI, Analysis Services, and Dataverse.

Specification for M: The document outlines the M language in a structured manner, starting with its lexical structure and progressively covering concepts like values, expressions, types, operators, functions, error handling, Let and If expressions, sections, and grammar.

Values in M: M has different value types, including primitive, list, record, and table values.

Functions: M supports functions, which are values that can produce new values when invoked with arguments.

Evaluation Model: The M language's evaluation model is similar to spreadsheet calculations, with dependencies determining the order of evaluation.

Metadata: Metadata records are used to associate additional information with values in a non-intrusive manner.

Let Expressions: Let expressions allow users to define auxiliary values for complex expressions, enhancing readability.

If Expressions: If expressions enable conditional evaluations based on logical conditions.

Error Handling: M provides error handling mechanisms, allowing users to gracefully manage errors that may occur during expression evaluation.

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 What is the primary purpose of Microsoft Power Query's "get data" feature?
   a) Data visualization
   b) Data transformation and retrieval
   c) Data encryption
   d) Data compression

2 Which programming language is informally known as "M" in the context of Microsoft Power Query?
   a) Python
   b) Java
   c) R
   d) M

3 In which Microsoft products can you find the integration of the Power Query formula language (M)?
   a) Word and PowerPoint
   b) Excel, Power BI, Analysis Services, and Dataverse
   c) Visual Studio
   d) Windows Operating System

4 What does the lexical structure define in the M formula language?
   a) Data types
   b) Text formats
   c) Valid texts
   d) Conditional statements

5 Which M language concept represents a set of name/value pairs and allows names to be written without quotes?
   a) Primitive values
   b) Functions
   c) Records
   d) Lists

6 What is the primary function of the Value.Metadata function in M?
   a) Define new values
   b) Access metadata records for values
   c) Create records with metadata
   d) Calculate mathematical expressions

7 What kind of values can be used as operands with operators in M expressions?
a) Only numbers
b) Only text values
c) Numbers, text values, and records
d) Only records

8 What is the primary role of Let expressions in M?
a) Define new functions
b) Handle errors
c) Create metadata records
d) Define auxiliary values for complex expressions

9 Which evaluation model in M is similar to spreadsheet calculations, where order is determined by dependencies?
a) Eager evaluation
b) Lazy evaluation
c) Spreadsheet evaluation
d) Conditional evaluation

10 How are errors managed in M expressions?
a) Errors are ignored
b) Errors are automatically corrected
c) Errors are handled using the try expression
d) Errors terminate the evaluation process

11 What is the primary purpose of If expressions in M?
a) Defining functions
b) Handling metadata
c) Conditional evaluation
d) Creating lists

12 In M, what are sections primarily used for?
a) Creating metadata
b) Defining functions
c) Organizing records
d) Modularity (not yet leveraged by Power Query)

13 What does the M formula language aim to achieve for computer language theorists?
a) Pure functional programming
b) Object-oriented programming
c) Imperative programming
d) Procedural programming

14 What does the Value.Metadata function allow you to access in M?
   a) Metadata records
   b) Records within lists
   c) Values within tables
   d) Functions within expressions

15 What does the try expression convert errors and proper values into in M?
   a) New expressions
   b) Metadata records
   c) Lazy evaluations
   d) Record values indicating error handling

Answers

1 *b) Data transformation and retrieval*
2 *d) M*
3 *b) Excel, Power BI, Analysis Services, and Dataverse*
4 *c) Valid texts*
5 *c) Records*
6 *b) Access metadata records for values*
7 *c) Numbers, text values, and records*
8 *d) Define auxiliary values for complex expressions*
9 *a) Eager evaluation*
10 *c) Errors are handled using the try expression*
11 *c) Conditional evaluation*
12 *d) Modularity (not yet leveraged by Power Query)*
13 *a) Pure functional programming*
14 *a) Metadata records*
15 *d) Record values indicating error handling*

SELF-EXAMINATION QUESTIONS FOR PRACTICE:

1 What is the core capability of Microsoft Power Query when it comes to data manipulation?

2 What is the informal name for the formula language used in Microsoft Power Query?

3 Name four Microsoft products that embed the Power Query formula language.

4 What are the basic concepts covered in the specification for the M formula language?

5 Explain the difference between values and expressions in the M language.

6 How are errors handled in M expressions, and what is the purpose of the try expression?

7 What is the role of Let expressions in M, and how do they enhance the language?

8 How does the evaluation model in M resemble calculations in spreadsheets like Excel?

9 What is the purpose of metadata records in M, and how are they associated with values?

10 How can If expressions be used in M, and what do they enable you to do within an expression?

CHAPTER 5

5.8. TABLE AND LIST FUNCTIONS

LEARNING OBJECTIVES

Upon completing this module on "Table and List Functions," learners will:

- Parameter Usage: Understand how to name output columns in tables and utilize parameters in functions like Table.FromRows and Table.FromList.
- Comparison Mastery: Grasp the sorting and ordering criteria, including numeric values, key selectors, comparer functions, and custom logic.
- Count and Condition Handling: Expertly handle row counts and filtering based on conditions.
- Manage Extra Values: Effectively handle extra values within rows using numeric parameters.
- Handle Missing Columns: Manage missing columns in operations using numeric parameters.
- Sort Proficiency: Apply sorting techniques using numeric parameters.
- Equation Criteria Application: Utilize equation criteria for tables and lists.
- Data Type Understanding: Identify and work with comparable data types.
- Occurrence Specification: Apply occurrence specifications in list operations.
- Replacement Operations: Implement replacements with old and new values.

5.8.1. TABLE FUNCTIONS:

Parameter Values: Naming output columns

This parameter is a list of text values specifying the column names of the resulting table. This parameter is generally used in the Table construction functions, such as Table.FromRows and Table.FromList.

Comparison criteria:

Comparison criterion can be provided as either of the following values:

- A number value to specify a sort order. More information: Sort order
- To compute a key to be used for sorting, a function of one argument can be used.
- To both select a key and control order, comparison criterion can be a list containing the key and order.
- To completely control the comparison, a function of two arguments can be used that returns -1, 0, or 1 given the relationship between the left and right inputs. Value.Compare can be used to delegate this logic.

For examples, go to the description of Table.Sort.

Count or Condition criteria:

This criterion is generally used in ordering or row operations. It determines the number of rows returned in the table and can take two forms, a number or a condition.

- A number indicates how many values to return in line with the appropriate function.
- If a condition is specified, the rows containing values that initially meet the condition is returned. Once a value fails the condition, no further values are considered.

Handling of extra values:

Extra values are used to indicate how the function should handle extra values in a row. This parameter is specified as a number, which maps to the following options:

ExtraValues.List = 0 ExtraValues.Error = 1 ExtraValues.Ignore = 2

Missing column handling:

This parameter is used to indicate how the function should handle missing columns. This parameter is specified as a number, which maps to the following options:

MissingField.Error = 0 MissingField.Ignore = 1 MissingField.UseNull = 2;

This parameter is used in column or transformation operations, for examples, in Table.TransformColumns. More information: MissingField.Type

Sort Order:

Sort ordering is used to indicate how the results should be sorted. This parameter is specified as a number, which maps to the following options:

Order.Ascending = 0 Order.Descending = 1

Equation criteria:

Equation criteria for tables can be specified as either:

A function value that is either:

- A key selector that determines the column in the table to apply the equality criteria.
- A comparer function that is used to specify the kind of comparison to apply. Built-in comparer functions can be specified. More information: Comparer functions

A list of the columns in the table to apply the equality criteria.

5.8.2. LIST FUNCTIONS

Ordering:

Ordering functions perform comparisons. All values that are compared must be comparable with each other. This means they must all come from the same datatype (or include null, which always compares smallest). Otherwise, an Expression.Error is thrown.

Comparable data types include:

- Number
- Duration
- DateTime
- Text
- Logical
- Null

Parameter values:

Occurrence specification

- Occurrence.First = 0;
- Occurrence.Last = 1;
- Occurrence.All = 2;

Sort order

- Order.Ascending = 0;
- Order.Descending = 1;

Equation criteria:

Equation criteria for list values can be specified as either:

A function value that is either:

- A key selector that determines the value in the list to apply the equality criteria.
- A comparer function that is used to specify the kind of comparison to apply. Built in comparer functions can be specified—go to Comparer functions.

A list value that has:

- Exactly two items.
- The first element is the key selector as specified above.
- The second element is a comparer as specified above.

Comparison criteria:

Comparison criterion can be provided as either of the following values:

- A number value to specify a sort order. For more information, go to Sort order.
- To compute a key to be used for sorting, a function of one argument can be used.
- To both select a key and control order, comparison criterion can be a list containing the key and order.
- To completely control the comparison, a function of two arguments can be used that returns -1, 0, or 1 given the relationship between the left and right inputs. Value.Compare is a method that can be used to delegate this logic.

Replacement operations:

Replacement operations are specified by a list value. Each item of this list must be:

- A list value of exactly two items.
- First item is the old value in the list, to be replaced.
- Second item is the new value, which should replace all occurrences of the old value in the list.

SUMMARY

Table Functions:

- Parameter Values: Learn to specify column names for resulting tables, primarily used in Table construction functions.
- Comparison Criteria: Understand various options for sorting and ordering, including numeric values, key selectors, comparer functions, and custom comparison logic.
- Count or Condition Criteria: Explore how to determine the number of rows returned in a table based on numeric values or conditions.
- Handling Extra Values: Learn to handle extra values in rows using numeric parameters.
- Missing Column Handling: Understand how to manage missing columns in operations.
- Sort Order: Discover how to specify sorting preferences using numeric parameters.
- Equation Criteria: Learn about applying equality criteria to tables, including key selectors and comparer functions.

List Functions:

- Ordering: Explore functions for comparing values within lists, ensuring compatibility among data types.
- Parameter Values: Understand occurrence specifications and sort order preferences using numeric parameters.
- Equation Criteria: Apply equality criteria to list values, utilizing key selectors, comparer functions, and custom comparisons.
- Comparison Criteria: Learn various methods for specifying comparisons, including numeric values and custom logic.
- Replacement Operations: Perform replacements in lists, specifying old and new values.

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 What is the primary purpose of the "Parameter Values" parameter in table functions?
   a) Sorting data
   b) Naming output columns
   c) Filtering rows
   d) Handling missing columns

2 In table functions, what are the various options for specifying the sort order in the "Comparison Criteria"?
   a) Numeric values, key selectors, comparer functions, and custom logic
   b) Text values only
   c) Date and time values only
   d) Logical values only

3 How is the count of rows to be returned determined in table functions that use "Count or Condition Criteria"?
   a) It is determined randomly
   b) It is specified as a condition
   c) It is specified as a numeric value
   d) It depends on the number of columns in the table

4 What does the "Handling of extra values" parameter in table functions indicate?
   a) How to handle missing values
   b) How to handle values that exceed a certain limit
   c) How to handle extra values in a row
   d) How to handle duplicate values

5 When and where is the "Missing Column Handling" parameter typically used in table functions?
   a) In row operations
   b) In sorting operations
   c) In column or transformation operations
   d) In list functions only

6 What are the two options mapped by the "Sort Order" parameter in table functions?
   a) Ascending and descending
   b) Random and sequential
   c) Increasing and decreasing
   d) Odd and even

7 In the context of table functions, what are "Equation Criteria," and how can they be specified?
   a) Criteria for mathematical equations
   b) Criteria for sorting tables
   c) Criteria for filtering rows
   d) Criteria for equality comparisons, specified using key selectors or comparer functions

8 Which of the following data types is NOT considered a comparable data type in ordering functions for lists?
   a) Number
   b) Text
   c) Logical
   d) Currency

9 What is the purpose of the "Occurrence Specification" parameter in list functions, and what are its possible values?
   a) It specifies column names in lists (values, keys)
   b) It determines the occurrence of specific values in lists
   c) It sets the order of sorting in lists
   d) It specifies how many occurrences to consider (First, Last, All)

10 How are replacement operations defined in list functions, and what should each item in the replacement list contain?
   a) Defined by specifying the old and new values; each item in the list contains two values
   b) Defined by specifying the old and new columns; each item in the list contains column names
   c) Defined by specifying numeric values; each item in the list contains a single number
   d) Defined by specifying sorting criteria; each item in the list contains sorting rules

11 What is the purpose of the "Comparison Criteria" in table functions?
   a) To specify column names
   b) To control the order of rows
   c) To determine the number of rows
   d) To define how values are compared and sorted

12 Which of the following is NOT a possible value for the "Handling of extra values" parameter in table functions?
   a) ExtraValues.List
   b) ExtraValues.Error
   c) ExtraValues.Skip
   d) ExtraValues.Ignore

13 In list functions, what is the primary purpose of the "Sort Order" parameter?
   a) To control the order of rows
   b) To specify column names
   c) To indicate how the results should be sorted
   d) To filter rows based on conditions

14 What are the key factors that determine whether values are comparable in list functions?
   a) Values must belong to the same datatype and cannot include null
   b) Values must be numeric and not include null
   c) Values must be sorted in ascending order
   d) Values must include null to be comparable

15 In replacement operations within list functions, what does each item in the replacement list contain?
   a) A single value to be replaced
   b) A key selector for sorting
   c) Two items: the old value to be replaced and the new value to replace it
   d) A condition to determine replacements

Answers

1 *B. Naming output columns*
2 *A. Numeric values, key selectors, comparer functions, and custom logic*
3 *C. It is specified as a numeric value*
4 *C. How to handle extra values in a row*
5 *C. In column or transformation operations*
6 *A. Ascending and descending*
7 *D. Criteria for equality comparisons, specified using key selectors or comparer functions*
8 *D. Currency*
9 *D. It specifies how many occurrences to consider (First, Last, All)*
10 *A. Defined by specifying the old and new values; each item in the list contains two values*
11 *D. To define how values are compared and sorted*
12 *C. ExtraValues.Skip*
13 *C. To indicate how the results should be sorted*
14 *A. Values must belong to the same datatype and cannot include null*
15 *C. Two items: the old value to be replaced and the new value to replace it*

SELF-EXAMINATION QUESTIONS FOR PRACTICE:

1 What is the primary purpose of the "Parameter Values" parameter in table functions?

2 In table functions, what are the various options for specifying the sort order in the "Comparison Criteria"?

3 How is the count of rows to be returned determined in table functions that use "Count or Condition Criteria"?

4 Explain the significance of the "Handling of extra values" parameter in table functions and provide the options it maps to.

5 When and where is the "Missing Column Handling" parameter typically used in table functions?

6 What are the two options mapped by the "Sort Order" parameter in table functions?

7 In the context of table functions, what are "Equation Criteria," and how can they be specified?

8 List the comparable data types that can be used in ordering functions for lists.

9 What is the purpose of the "Occurrence Specification" parameter in list functions, and what are its possible values?

10 How are replacement operations defined in list functions, and what should each item in the replacement list contain?

CHAPTER 5

5.9. CLEAN AND TRANSFORM DATA WITH POWER QUERY EDITOR

LEARNING OBJECTIVES

Upon completion of this section, you will be able to effectively utilize Microsoft Power BI Desktop's Power Query Editor to shape, adjust, and combine data from various sources by performing the following tasks:

- Apply data transformation techniques such as renaming columns and tables, changing data types, removing rows, and setting headers to meet specific data requirements within Power Query Editor.
- Proficiently navigate and utilize the extensive shortcut menus (right-click or context menus) in Power Query Editor to perform data transformation tasks efficiently.
- Record and utilize the sequence of applied steps in Power Query Editor to consistently shape data every time it's loaded, ensuring data remains in the desired format.
- Connect to external data sources, such as web resources, and import data into Power Query Editor, effectively shaping and preparing it for analysis.
- Filter and remove unnecessary data from imported datasets, such as removing rows, filtering specific values, and eliminating unneeded columns.
- Renaming columns and tables both within Power Query Editor and by changing their names in the Query Settings pane.
- Merging and appending queries and will be able to effectively combine data from multiple sources by merging queries, selecting matching columns, and expanding tables as needed.
- Apply the changes made in Power Query Editor and load the transformed data into Power BI Desktop, ensuring it's ready for further analysis and visualization.
- Grasp the importance of well-prepared data for reporting and visualization within Power BI Desktop, setting the stage for creating insightful reports.
- Save Power BI Desktop files, ensuring they can revisit and continue their work in subsequent sessions.

Now that we've connected to a data source by using Microsoft Power BI Desktop, we must adjust the data to meet our needs. Sometimes, adjusting means transforming the data by, for example, renaming columns or tables, changing text to numbers, removing rows, or setting the first row as a header.

Power Query Editor in Power BI Desktop makes extensive use of shortcut menus (also known as right-click or context menus), in addition to having tasks available on the ribbon. Most of what you can select on the Transform tab on the ribbon is also available by right-clicking an item (like a column) and then selecting a command on the shortcut menu that appears.

Shape data:

When you shape data in Power Query Editor, you're providing step-by-step instructions that Power Query Editor will carry out to adjust the data as it loads and presents it. The original data source isn't affected. Only this particular view of the data is adjusted, or shaped.

The steps you specify (for example, rename a table, transform a data type, or delete columns) are recorded by Power Query Editor. Those steps are then carried out each time the query connects to the data source, so that the data is always shaped the way you specify. This process occurs whenever you use the query in Power BI Desktop, or whenever anyone else uses your shared query (for example, in the Power BI service). The steps are captured sequentially under Applied Steps in the Power Query Settings pane.

The following image shows the Query Settings pane for a query that has been shaped. We'll go through each of the steps in the next few paragraphs.

Figure 5.9.1.: Query Settings

Let's return to the retirement data that we found by connecting to a Web data source, and let's shape that data to fit our needs.

We need the data to be numbers. They are fine in this case but if you ever need to change the data type, just right-click the column header, and then select Change Type > Whole Number. If you must change more than one column, select one of them, and then hold down the Shift key while you select additional adjacent columns. Then right-click a column header to change all the selected columns. You can also use the Ctrl key to select non-adjacent columns.

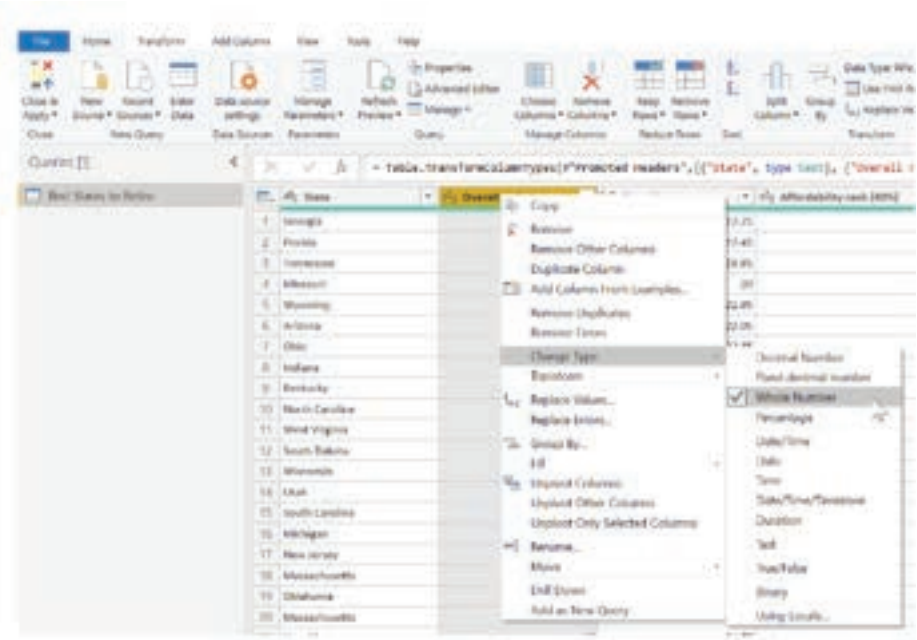


Figure 5.9.2.: Applied step changed type

Note:

Often, Power Query will detect that a column of text should be numbers, and will automatically change the data type when it brings the table into Power Query Editor. In this case, a step under Applied steps identifies what Power Query did for you.

You can also change, or transform, those columns from text to header by using the Transform tab on the ribbon. The following image shows the Transform tab. The red box highlights the Data Type button, which lets you transform the current data type to another.

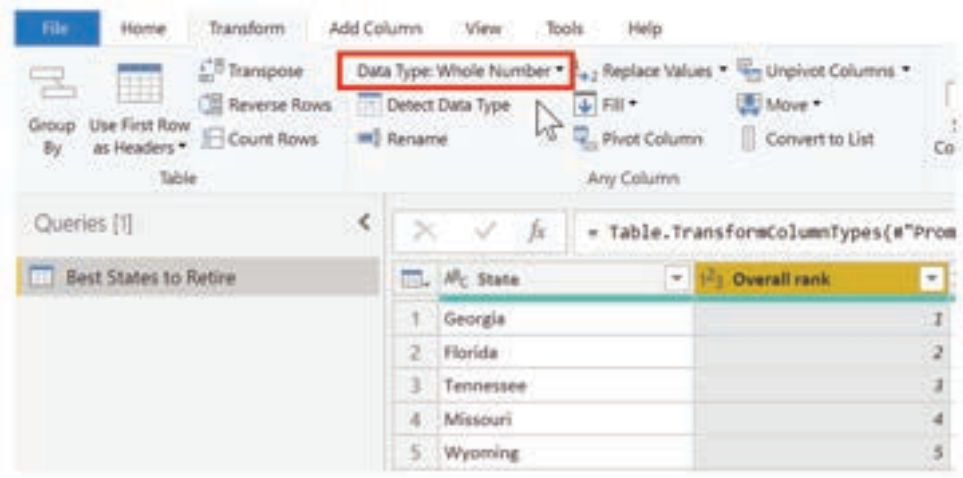


Figure 5.9.3.: The Transform ribbon and the Data Type button

Notice that the Applied Steps list in the Query Settings pane reflects all the changes that were made. To remove any step from the shaping process, just select it, and then select the X to the left of it.

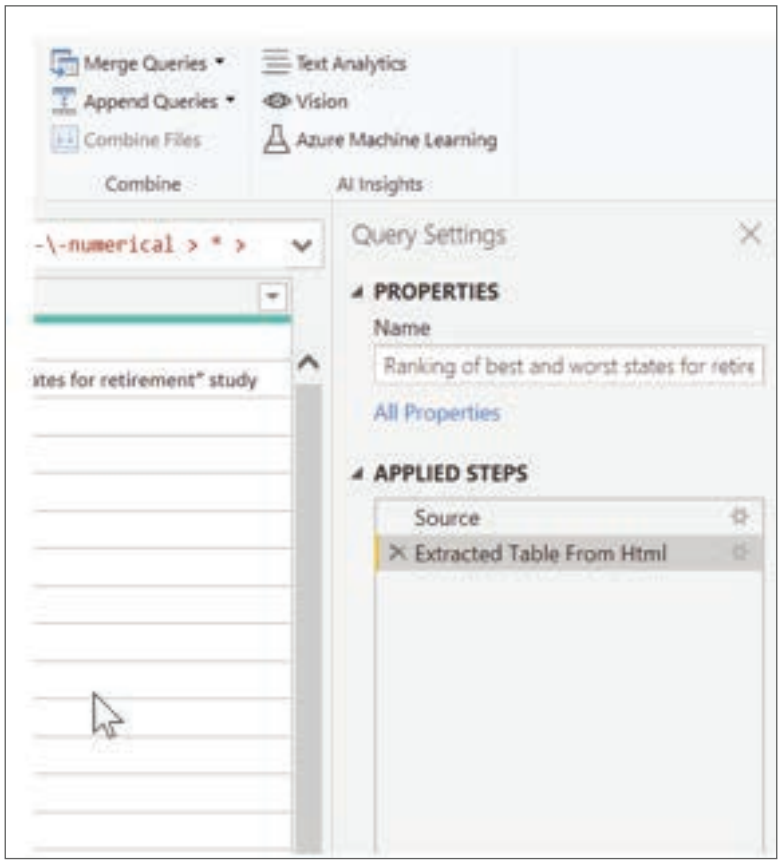


Figure 5.9.4.: Query Settings window

Connect to data:

That data about different states is interesting, and will be useful for building additional analysis efforts and queries. But there's one problem: most data out there uses a two-letter abbreviation for state codes, not the full name of the state. Therefore, we need some way to associate state names with their abbreviations.

We're in luck: there's another public data source that does just that, but it needs a fair amount of shaping before we can connect it to our retirement table. Here's the web resource for state abbreviations:

http://en.wikipedia.org/wiki/List_of_U.S._state_abbreviations

In Power Query Editor, on the Home tab on the ribbon, select New Source > Web. Then enter the address, and select OK. The Navigator window shows what it found on that webpage.

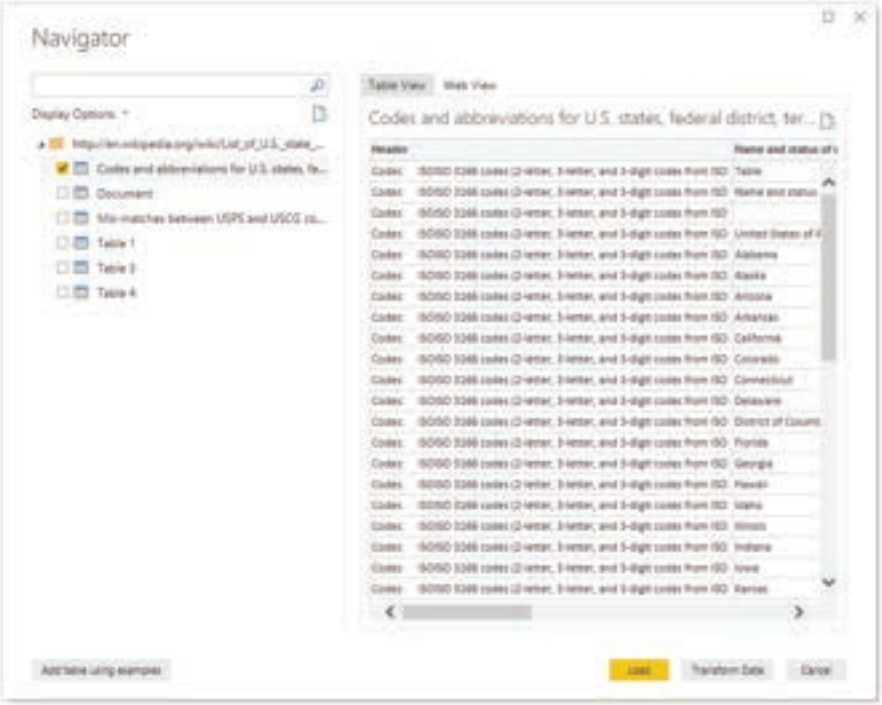


Figure 5.9.5.: US State abbreviations from website

Select the Codes and abbreviations... table, because it includes the data we want, although it's going to take quite a bit of shaping to pare down that data.

Select Load to bring the data into Power Query Editor so that we can shape it. Then follow these steps:

Remove the top three rows – Those rows are a result of the way the webpage's table was created, and we don't need them. To remove them, on the Home tab on the ribbon, select Remove rows > Remove Top Rows. In the dialog box that appears, enter 3 as the number of rows to remove.

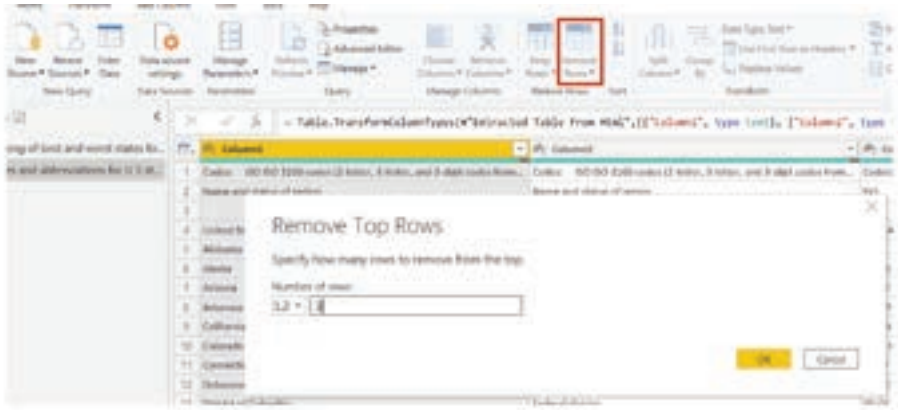


Figure 5.9.6.: Remove top rows

Remove the bottom 26 rows – Those rows are all for territories, which we don't need to include. The process is the same, but this time, select Remove rows > Remove Bottom Rows, and enter 26 as the number of rows to remove.

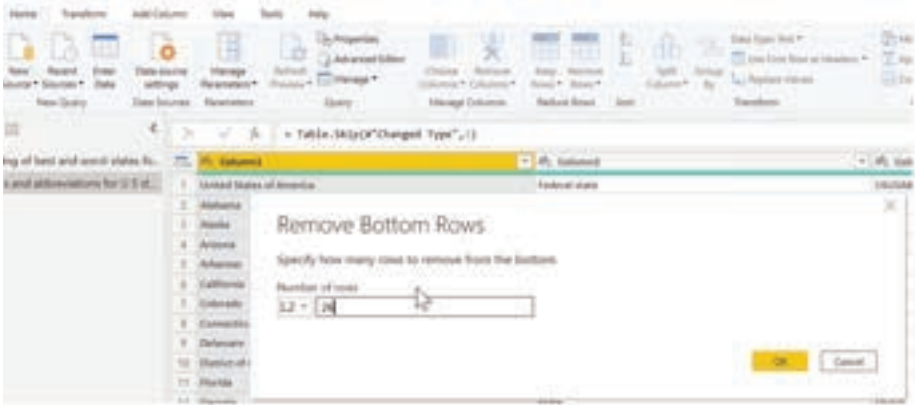


Figure 5.9.7.: Remove bottom rows

Filter out Washington, DC – The retirement stats table doesn't include Washington, DC, so we'll exclude it from our list. Select the drop-down arrow beside the Federal state column, and then clear the Federal district check box.

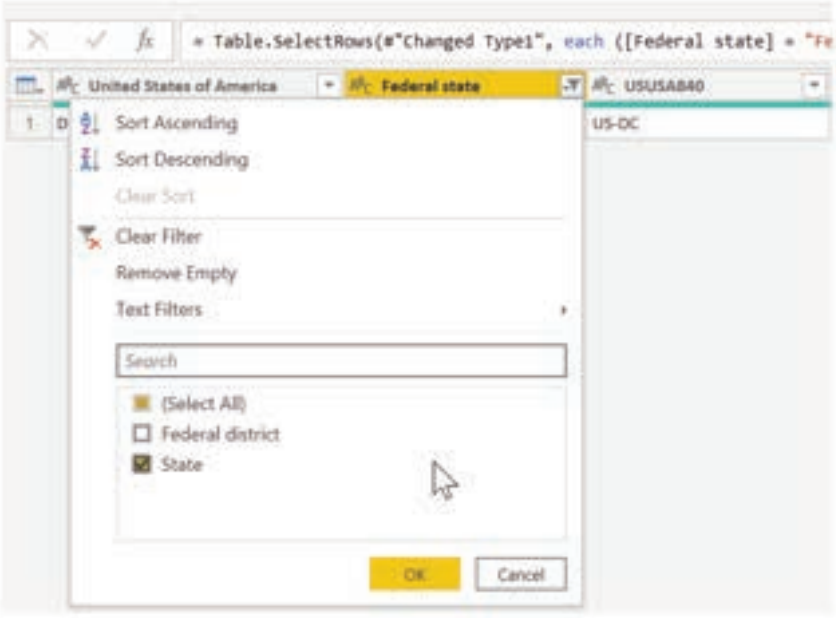


Figure 5.9.8.: Remove a row having a certain value

Remove a few unneeded columns – We just need the mapping of each state to its official two-letter abbreviation, and that information is given in the first and fourth columns. Therefore, we just need to keep those two columns and can remove all the others. Select the first column to remove, and then hold down the Ctrl key while you select the other columns to remove (this lets you select multiple, non-adjacent columns). Then, on the Home tab on the ribbon, select Remove Columns > Remove Columns.

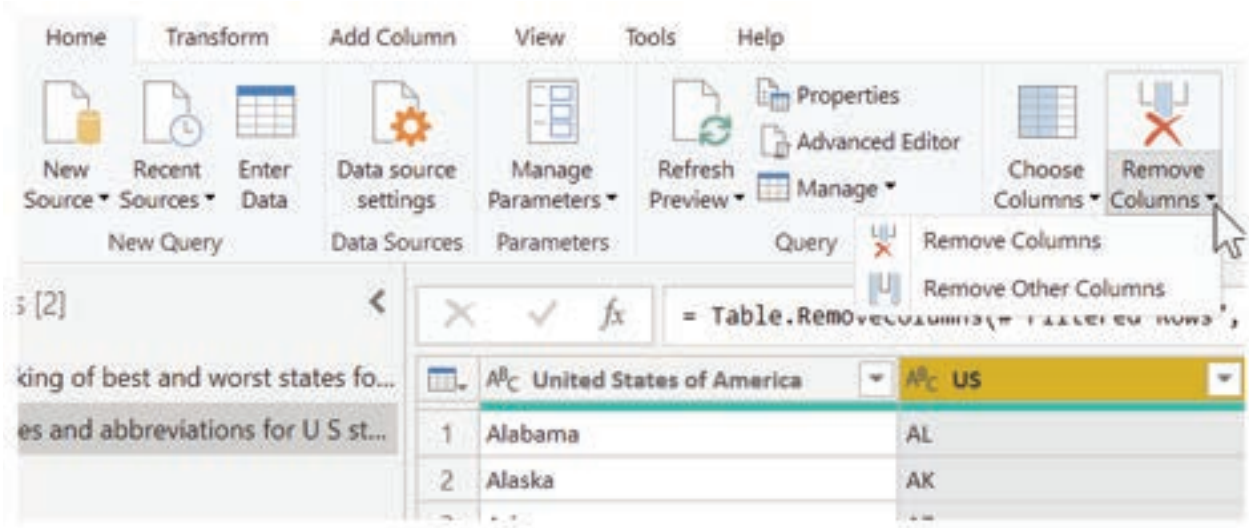


Figure 5.9.9.: Remove particular columns

Use the first row as headers – Because we removed the top three rows, the current top row is the header we want. Select the Use first row as headers button.

Figure 5.9.10.: Use first row as headers

Note:

This is a good time to point out that the sequence of applied steps in Power Query Editor is important and can affect how the data is shaped. It's also important to consider how one step might affect another subsequent step. If you remove a step from the Applied Steps list, subsequent steps might not behave as originally intended, because of the impact of the query's sequence of steps.

Rename the columns and the table itself – As usual, there are a couple ways to rename a column. You can use whichever way you prefer. Let's rename them State Name and State Code. To rename the table, just enter the name in the Name field in the Query Settings pane. Let's call this table StateCodes.

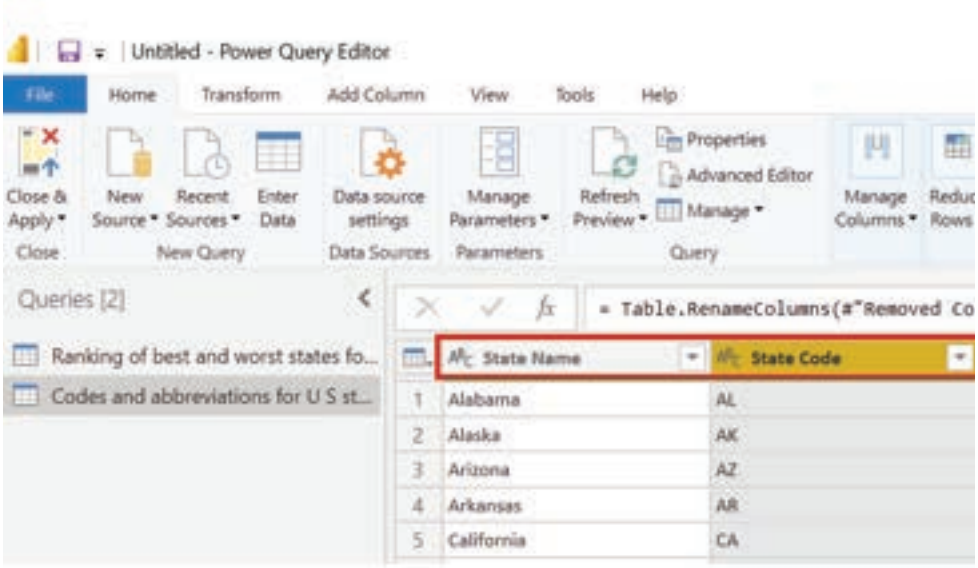


Figure 5.9.11.: Rename columns

Combine data:

Now that the StateCodes table is shaped, we can combine our two tables into one. Because the tables that we now have are a result of the queries we applied to the data, they're often referred to as queries.

There are two primary ways of combining queries: merging and appending.

When you have one or more columns that you want to add to another query, you merge the queries. When you have additional rows of data to add to an existing query, you append the query.

In this case, we want to merge the queries. To get started, select the query to merge the other query into. Then, on the Home tab on the ribbon, select Merge Queries. We want to select our retirement query first. While we're at it, let's rename that query RetirementStats.

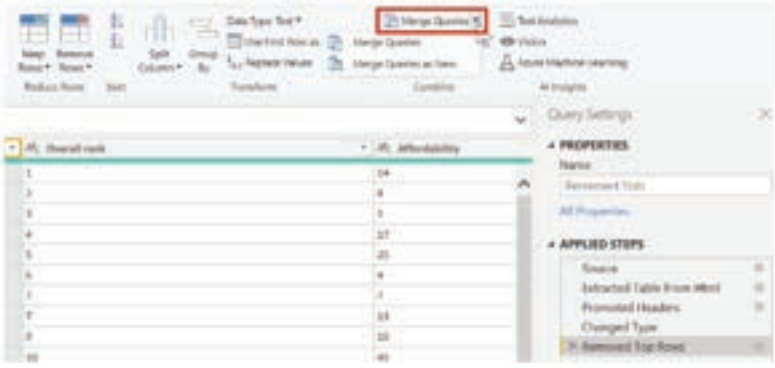


Figure 5.9.12.: Merge queries button

The Merge dialog box appears, prompting us to select the table to merge into the selected table, and the matching columns to use for the merge.

Select State from the RetirementStats table (query), and then select the StateCodes query. (In this case, the choice is easy, because there's only one other query. But when you connect to many data sources, there will be many queries to choose from.) After you select the correct matching columns—State from RetirementStats and State Name from StateCodes—the Merge dialog box will look like this, and the OK button will become available.

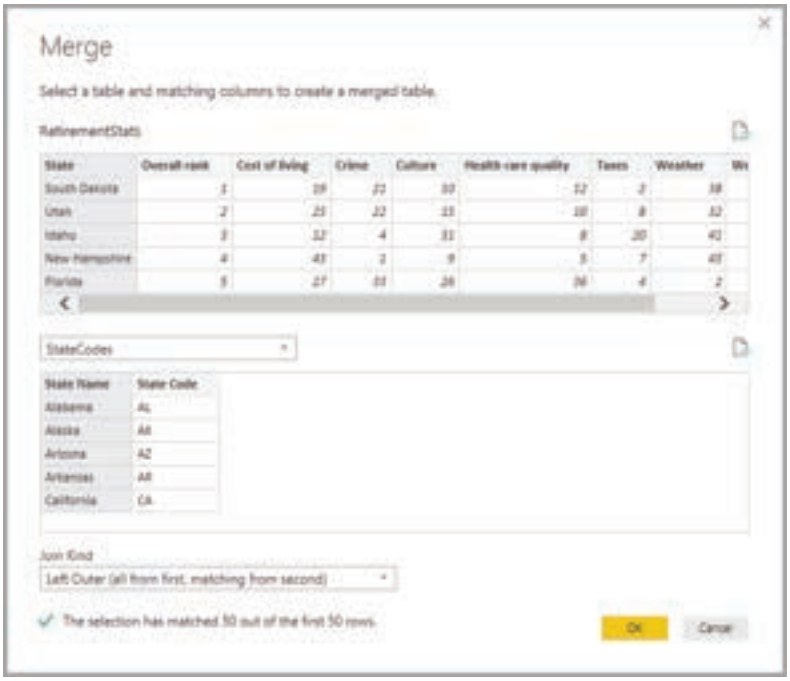


Figure 5.9.13.: Merge dialog box

A NewColumn is created at the end of the query and is the contents of the table (query) that was merged with the existing query. All columns from the merged query are condensed into the NewColumn, but you can expand the table and include whichever columns you want. To expand the merged table and select the columns to include, select the expand icon (expand icon). The Expand dialog box appears.

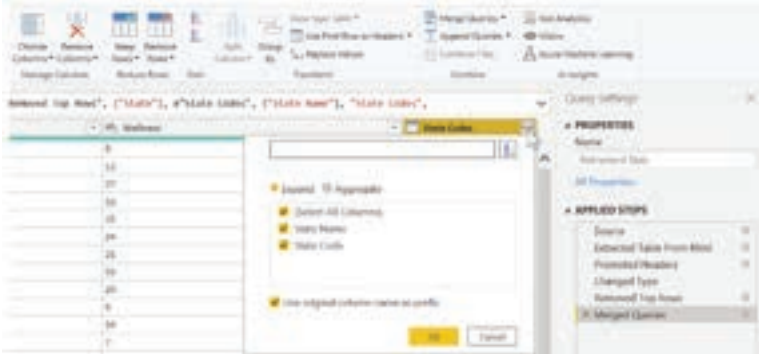


Figure 5.9.14.: Expand dialog box

In this case, we just want the State Code column. Therefore, select only that column, and then select OK. You can also clear the Use original column name as prefix check box. If you leave it selected, the merged column will be named NewColumn.State Code (the original column name, or NewColumn, then a dot, and then the name of the column that's being brought into the query).

Note:

If you want, you can play around with how the NewColumn table is brought in. If you don't like the results, just delete the Expand step from the Applied Steps list in the Query Settings pane. Your query will return to the state it was in before you applied that step. It's like a free do-over that you can do as many times as you want, until the expand process looks the way you want.

We now have a single query (table) that combines two data sources, each of which has been shaped to meet our needs. This query can serve as a basis for lots of additional, interesting data connections, like housing cost statistics, demographics, or job opportunities in any state.

To apply the changes in Power Query Editor and load them into Power BI Desktop, select Close & Apply on the Home tab on the ribbon.

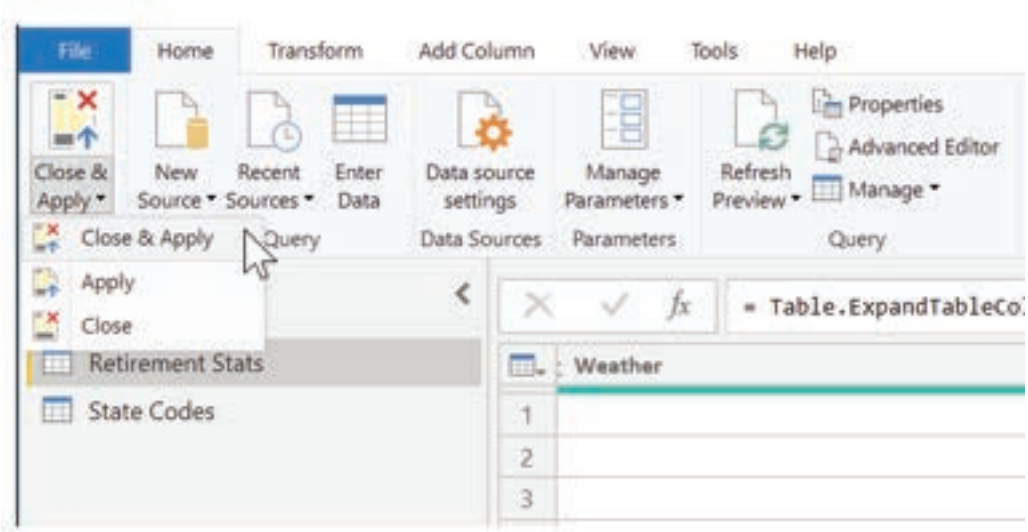


Figure 5.9.15.: Close and apply data settings

The data in your model is now ready to work with. Next, we'll create some visuals for your report.

For now, we have enough data to create a few interesting reports, all in Power BI Desktop. Because this is a milestone, let's save this Power BI Desktop file. Select File > Save on the Home tab on the ribbon to save the report—we'll call it Getting Started with Power BI Desktop.

SUMMARY

- Installation and Setup: Users can install Power BI Desktop from the Power BI service or Microsoft Store. Once installed, it runs as an application with three main views: Report, Data, and Model.
- Connecting to Data Sources: Users can connect to various data sources, such as web data, by selecting "Get Data" and specifying the data source URL. Power Query Editor allows for data transformation and refinement.
- Data Shaping with Power Query Editor: Power Query Editor helps users shape data by renaming columns, changing data types, removing rows, and more. Applied steps are recorded, ensuring data is consistently shaped when loaded.
- Combining Data Sources: Users can merge queries to combine data sources. This can be done by selecting "Merge Queries" and defining matching columns. The merged data can be expanded to include the desired columns.
- Saving and Applying Changes: Once data is shaped and merged, users can save the Power BI Desktop file. The changes can be applied by selecting "Close & Apply" in Power Query Editor.

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 What is the primary purpose of shaping data in Power Query Editor?
   a) To modify the original data source
   b) To adjust the data for presentation
   c) To delete the original data
   d) To create a new data source

2 Where are the steps for shaping data recorded in Power Query Editor?
   a) In the original data source
   b) In the Applied Steps list
   c) In the Transform ribbon
   d) In the Navigator window

3 How can you change the data type of a column in Power Query Editor?
a) Right-click the column header and select "Delete"
b) Select the column and press Ctrl+C
c) Right-click the column header and select "Change Type"
d) Click the column header and press Enter

4 What is the purpose of the "Transform" tab in Power Query Editor?
a) To delete columns
b) To rename tables
c) To change data types
d) To filter rows

5 How can you remove a step from the shaping process in Power Query Editor?
a) Right-click the step and select "Delete"
b) Press Ctrl+Z
c) Use the Ctrl key to select the step and press Delete
d) Select the step and press Enter

6 What is the purpose of the "New Source > Web" option in Power Query Editor?
a) To delete web data
b) To add web data as a data source
c) To rename a web data source
d) To change the data type of web data

7 How do you remove the top three rows from a table in Power Query Editor?
a) Right-click the rows and select "Delete"
b) Use the Ctrl key to select the rows and press Delete
c) Select the rows and press Enter
d) Use the "Remove rows > Remove Top Rows" option

8 What is the purpose of filtering out Washington, DC in Power Query Editor?
a) To add it to the list
b) To exclude it from the list
c) To change its data type
d) To rename it

9 How do you rename a column in Power Query Editor?
a) Right-click the column header and select "Rename"
b) Select the column and press Ctrl+R
c) Use the "Rename Columns" option in the Transform tab
d) Use the "Use first row as headers" option

10 What are the two primary ways of combining queries in Power Query Editor?
a) Renaming and appending
b) Merging and deleting
c) Shaping and transforming
d) Merging and appending

11 When should you use the "Merge Queries" option in Power Query Editor?
a) To add rows to an existing query
b) To rename columns in a query
c) To combine data from different sources
d) To delete queries

12 What is the purpose of the "Expand" step when merging queries?
a) To delete rows
b) To combine columns
c) To include selected columns from the merged query
d) To change data types

13 How can you undo an unwanted expansion of a merged table in Power Query Editor?
a) Press Ctrl+Z
b) Delete the merged table
c) Remove the "Expand" step from the Applied Steps list
d) Close and reopen Power Query Editor

14 What is the final step to apply changes in Power Query Editor and load them into Power BI Desktop?
a) Close and Apply
b) Delete and Discard
c) Save and Exit
d) Reset and Reopen

15 In Power BI Desktop, which of the following is NOT one of the three main views?
a) Report
b) Data
c) Model
d) Query

Answers

1 *b) To adjust the data for presentation*
2 *b) In the Applied Steps list*
3 *c) Right-click the column header and select "Change Type"*
4 *c) To change data types*
5 *a) Right-click the step and select "Delete"*
6 *b) To add web data as a data source*
7 *d) Use the "Remove rows > Remove Top Rows" option*
8 *b) To exclude it from the list*
9 *c) Use the "Rename Columns" option in the Transform tab*
10 *d) Merging and appending*
11 *c) To combine data from different sources*
12 *c) To include selected columns from the merged query*
13 *c) Remove the "Expand" step from the Applied Steps list*
14 *a) Close and Apply*
15 *d) Query*

SELF-EXAMINATION QUESTIONS FOR PRACTICE:

1 What are the three main steps involved in working with Power BI Desktop?

2 What is the file format used to save work in Power BI Desktop, and how can it be shared?

3 Where can you download Power BI Desktop, and what are the alternative installation options?

4 Describe the three main views available in Power BI Desktop and how to switch between them.

5 How do you connect to data sources in Power BI Desktop, and what types of data sources can you connect to?

6 What is Power Query Editor, and how does it help with data transformation?

7 Explain the concept of "shaping data" in Power BI Desktop and why it is important.

8 What are "applied steps" in Power Query Editor, and how do they impact data shaping?

9 What is the process of combining two queries in Power BI Desktop, and what are the primary ways to do it?

10 How can you save your work in Power BI Desktop, and why is it important to do so?

CHAPTER 5

5.10. OUTPUT OPTIONS IN POWER BI

LEARNING OBJECTIVES

By the end of this training, you will be proficient in using Power BI's diverse output options for effective report sharing and distribution. They will be able to:

- Identify and comprehend the primary Power BI output methods, including publishing to Power BI Service, exporting to PDF, PowerPoint, and Excel, exporting data, publishing to the web, setting up email subscriptions, printing reports, using Power BI Embedded for custom applications, and sharing .PBIX files.
- Demonstrate how to publish Power BI reports to the Power BI Service for interactive access, collaboration, and automated data refresh.
- Explain how to export reports to PDF and PowerPoint for static snapshots and presentation purposes.
- Guide users in exporting visual data to Excel or CSV for further analysis.
- Describe the "Publish to web" option and its implications.
- Instruct users on exporting summarized data to Excel.
- Provide guidance on setting up email subscriptions.
- Explain how to print Power BI reports.
- Introduce Power BI Embedded for embedding reports in custom applications.
- Clarify how to share .PBIX files for collaborative work.

Power BI offers several output options to help you share and distribute your reports and insights effectively. These output options cater to various use cases and user preferences. Here are the primary output options in Power BI:

Publish to Power BI Service:

This is one of the most common ways to share your Power BI reports. You can publish your report to the Power BI cloud service, making it accessible to others who have access to your workspace. Key features include:

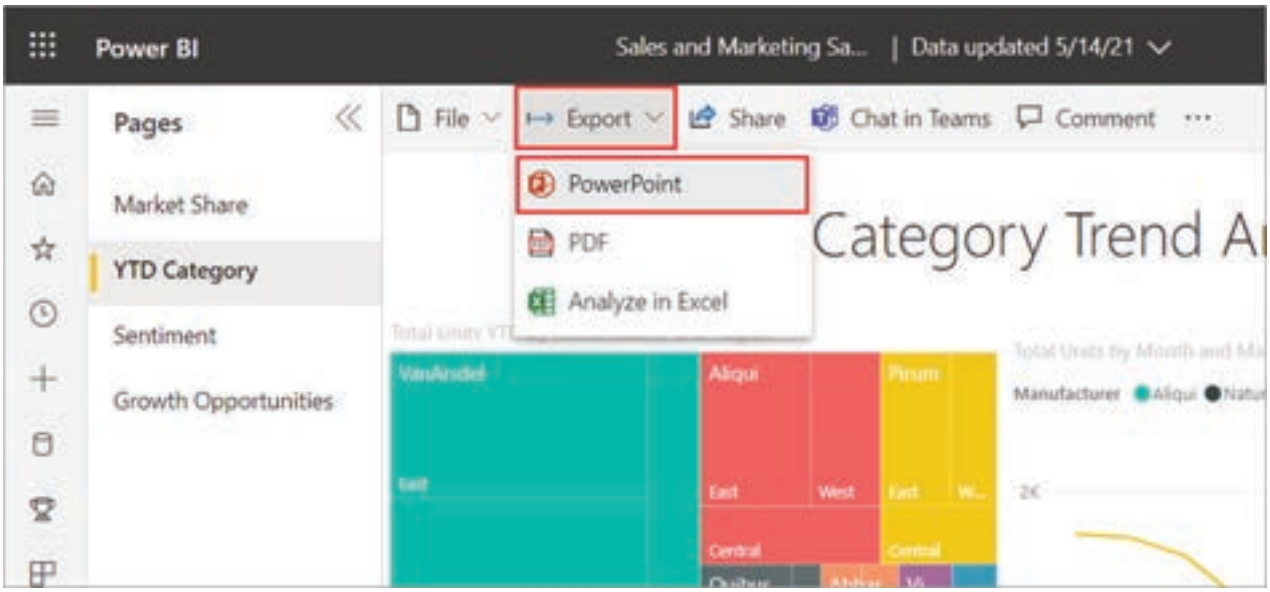
- Interactive access to the report via a web browser.
- Collaboration and sharing with other Power BI users.
- Ability to set up automatic data refresh schedules.

Export to PDF:

You can export your Power BI report as a PDF document. This is useful for creating static snapshots of your report that can be easily shared, printed, or archived.
To export to PDF, go to "File" > "Export" > "PDF."

Export to PowerPoint:

Exporting to PowerPoint generates a PowerPoint presentation with slides based on your report visuals. This is handy for creating slide decks for presentations.
To export to PowerPoint, go to "File" > "Export" > "PowerPoint."



Export Data:

You can export the underlying data from a visual in your report to Excel or CSV format. This is useful when users want to perform further analysis on the data.
To export data, select the visual, go to "More options" (represented by three dots) on the visual, and choose "Export data."

Publish to Web:

If you want to share your Power BI report publicly or embed it in a website or blog, you can use the "Publish to web" option. Be cautious with this option, as it makes your report accessible to anyone on the internet.
To use "Publish to web," go to "File" > "Publish to web."

Export to Excel:

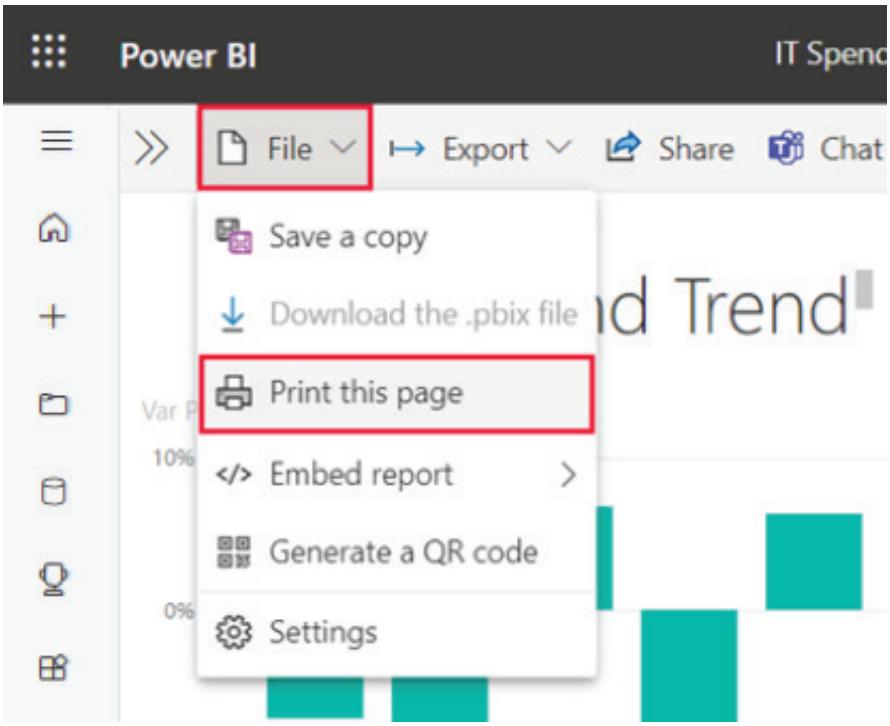
You can export summarized data from visuals to Excel. This is beneficial for users who prefer working with data in Excel or want to perform further analysis.
To export to Excel, select the visual, go to "More options," and choose "Analyze in Excel."

Email Subscription:

Power BI allows you to set up email subscriptions for your reports. Users can receive regular email updates with a link to the report or a PDF attachment.
To set up email subscriptions, click "Subscribe" within the Power BI Service.

Print Reports:

You can print your report directly from the Power BI Service or the Power BI Desktop application. This is useful for creating physical copies for meetings or documentation.
To print a report, use the print option available in Power BI.



Power BI Embedded:

For embedding Power BI reports into custom applications, websites, or portals, you can use Power BI Embedded. It allows developers to integrate Power BI reports seamlessly into their applications for broader distribution.
Power BI Embedded requires development and Azure resources.

SUMMARY

- Publish to Power BI Service: This option enables you to share your Power BI reports by publishing them to the Power BI cloud service. It offers interactive access through a web browser, collaboration with other Power BI users, and the ability to schedule automatic data refreshes.
- Export to PDF: You can export your Power BI report as a PDF document, creating static snapshots suitable for sharing, printing, or archiving. The export process is accessible through the "File" menu.
- Export to PowerPoint: This feature generates a PowerPoint presentation based on your report visuals, making it convenient for creating slide decks for presentations. It can be accessed from the "File" menu.
- Export Data: Users can export underlying data from a visual in your report to Excel or CSV format. This is beneficial for those who want to perform further data analysis. The export data option is available through the visual's context menu.
- Publish to Web: For public sharing or embedding in websites and blogs, the "Publish to web" option allows you to make your Power BI report accessible to internet users. However, caution is advised due to its public accessibility. You can access this option through the "File" menu.
- Export to Excel: This option permits the export of summarized data from visuals to Excel, facilitating further data manipulation. It is accessible through the visual's context menu.
- Email Subscription: Power BI offers the ability to set up email subscriptions for reports. Users can receive regular email updates with links to the report or PDF attachments. The subscription setup is available within the Power BI Service.
- Print Reports: Reports can be directly printed from both the Power BI Service and Power BI Desktop application. This is advantageous for creating physical copies for meetings or documentation purposes.
- Power BI Embedded: Developers can use Power BI Embedded to seamlessly integrate Power BI reports into custom applications, websites, or portals for broader distribution. However, this option requires development skills and Azure resources.
- File Distribution: Sharing the .PBIX file itself allows others to open and interact with the report using Power BI Desktop, provided they have the application installed.
- These output options in Power BI cater to a wide range of sharing and distribution needs, making it a versatile tool for disseminating insights and reports.

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 What is one of the key features of publishing a Power BI report to the Power BI Service?
   a) Creating static snapshots
   b) Collaboration and sharing with other users
   c) Exporting data to PowerPoint
   d) Embedding in custom applications

2 Which output option in Power BI is best suited for creating static snapshots of your report for sharing, printing, or archiving?
   a) Export to PDF
   b) Export to Excel
   c) Publish to Web
   d) Power BI Embedded

3 Which Power BI output option generates a PowerPoint presentation based on your report visuals for presentations?
   a) Publish to Power BI Service
   b) Export to PowerPoint
   c) Export Data
   d) Email Subscription

4 What is the purpose of exporting data from a visual in a Power BI report?
   a) Creating PowerPoint presentations
   b) Sharing with the public
   c) Further analysis in Excel or CSV
   d) Setting up email subscriptions

5 Which Power BI output option should you use when you want to share your report publicly or embed it in a website or blog?
   a) Export to Excel
   b) Publish to Web
   c) Print Reports
   d) File Distribution

6 What can users do with the "Publish to web" option in Power BI?
   a) Collaborate with others
   b) Share the .PBIX file
   c) Embed the report in custom applications
   d) Access the report publicly on the internet

7 Which Power BI output option is beneficial for users who prefer working with data in Excel or need to perform additional analysis?
   a) Export to PowerPoint
   b) Export Data
   c) Email Subscription
   d) Power BI Embedded

8 What can you do with the email subscription feature in Power BI?
   a) Export the report to PDF
   b) Set up automatic data refresh schedules
   c) Share the .PBIX file
   d) Receive regular email updates with a link to the report

9 What is the primary purpose of the "Print Reports" option in Power BI?
   a) Exporting the report to Excel
   b) Creating physical copies for meetings or documentation
   c) Sharing with the public
   d) Embedding in custom applications

10 What is Power BI Embedded mainly used for?
   a) Exporting reports to PDF
   b) Creating static snapshots
   c) Embedding Power BI reports into custom applications
   d) Collaborating with other users

11 When might you share the .PBIX file of your Power BI report with others?
   a) For setting up email subscriptions
   b) To publish to the Power BI Service
   c) To embed in a website
   d) For collaborative work using Power BI Desktop

12 Which output option allows you to create interactive access to your Power BI report via a web browser?
   a) Export to PDF
   b) Publish to Web
   c) Export Data
   d) File Distribution

13 What does Power BI's "Export to Excel" option allow you to do?
   a) Share your report on the internet
   b) Export underlying data from visuals to Excel
   c) Create PowerPoint presentations
   d) Set up email subscriptions

14 Which Power BI output option requires development and Azure resources for integration into custom applications?
   a) Publish to Power BI Service
   b) Export to Excel
   c) Power BI Embedded
   d) Print Reports

15 What is the primary advantage of exporting a Power BI report to the Power BI Service?
   a) Creating static snapshots
   b) Collaborative access and sharing
   c) Embedding in custom applications
   d) Exporting data to Excel

Answers

1 *B) Collaboration and sharing with other users*
2 *A) Export to PDF*
3 *B) Export to PowerPoint*
4 *C) Further analysis in Excel or CSV*
5 *B) Publish to Web*
6 *D) Access the report publicly on the internet*
7 *B) Export Data*
8 *D) Receive regular email updates with a link to the report*
9 *B) Creating physical copies for meetings or documentation*
10 *C) Embedding Power BI reports into custom applications*
11 *D) For collaborative work using Power BI Desktop*
12 *B) Publish to Web*
13 *B) Export underlying data from visuals to Excel*
14 *C) Power BI Embedded*
15 *B) Collaborative access and sharing*

SELF-EXAMINATION QUESTIONS FOR PRACTICE:

1 What are the key benefits of publishing a Power BI report to the Power BI Service as opposed to other output options?

2 How can you export a Power BI report as a PDF document, and what are the typical use cases for this option?

3 Explain how exporting a Power BI report to PowerPoint can be useful in a business context.

4 In Power BI, what steps do you need to follow to export the underlying data from a visual to Excel format?

5 What precautions should you take when considering the "Publish to web" option in Power BI, and what types of scenarios might it be suitable for?

6 How can you set up email subscriptions for Power BI reports, and what are the advantages of this feature?

7 Describe when it would be beneficial to print a Power BI report directly from the Power BI Service or Power BI Desktop.

8 What is Power BI Embedded, and for what purposes might developers choose to use it?

9 In which situations would sharing the .PBIX file itself be a preferred method of distributing a Power BI report?


UNIT
2
Python

CHAPTER 1
PYTHON FOR DATA SCIENCE

LEARNING OBJECTIVES

- Understanding the fundamentals of Python and its versatility.
- Understanding variables, their usage, and importance while coding.
- Different types of expressions and its use in arithmetic operations.
- Various methods of string operations.
- Properties of List and Tuple. Various operations performed on both List and Tuple.
- Conditioned-based use of loops like “if”, “for” and “while”.
- File handling and its manipulation.

1.1 INTRODUCTION TO PYTHON BASICS

Python is a high-level, versatile programming language. With the usage of extensive indentation, its design philosophy emphasizes code readability. Python uses garbage collection and has dynamic typing. It supports a variety of paradigms for programming, including structured programming, object-oriented programming, and functional programming.

In the world of programming, Python is one of the most widely used languages. It is employed in many different fields, such as artificial intelligence, machine learning, data science, and web development. Python is frequently used for activities involving automation and scripting.

Python is a robust and adaptable language that is simple to understand and use. Both novice and seasoned programmers will find it to be a fantastic option. In 1991, Guido van Rossum developed it, and it became available.

The advantages of Python are as follows: -

- Python runs on a variety of operating systems, including Windows, Mac, Linux, and Raspberry Pi.
- It is beginner beginner-friendly language which makes replaces the complex coding in other languages with a simple one.
- The syntax of Python is straightforward and resembles that of English.
- Python's syntax differs from various other programming languages in that it enables programmers to construct applications with fewer lines of code.
- Python operates on an interpreter system, allowing for the immediate execution of written code. As a result, prototyping can proceed quickly.
- Python can be used in a functional, object-oriented, or procedural manner.

Python can be applied in the following ways: -

- Web applications can be developed on a server using Python.
- Workflows can be made with Python and other technologies.
- Database systems are connectable with Python. Files can also be read and changed by it.
- Big data management and advanced mathematical operations can both be done with Python.
- Python can be used to produce software that is ready for production or rapid prototyping.
- Python is one of the most user-friendly languages for Data Science. The in-built libraries in Python make users easily code and get the required output.

1.1.1 VARIABLES AND EXPRESSIONS

Variables

The ability to alter variables is one of a programming language's most potent capabilities. A variable is a term used to identify a value. There is no command in Python for declaring variables. Variables can take any values from numbers to text or a combination of both. When a variable is initially given a value, it is considered to have been formed. Variables can change their type after they are set and are not required to be declared with a certain type. Initially, the type will be decided by Python only, which the user can change afterward. A variable is assigned a value in the assignment statement:

```
>> message = 'Hellow World'
>> n = 5
>> pi = 3.14
```

Three assignments are made in this case. In the first, a “message” variable is given the string value "Hello World”, The second allocates the integer “5” to “n”, and the third provides a variable called “pi” the floating-point number “3.14”.

We "remember" things using variables in computer programs, such as the score of the current football game. However, variables can change. This indicates that they are dynamic, much like the scoreboard during a football game, and can vary over time. A variable may initially have one value assigned to it, followed by another. (This differs from mathematics. If you input the value 3 for 'x' in math, it cannot change to link to a different value mid-calculation! The following example will help to understand it better.

```
>>> name = 'abc'
>>> abc
'abc'

>>> name = 25
>>> mno
25

>>> name = '25xyz'
>>> xyz
'25xyz'
```

Here name variable is changed three times. Every time we used different values to assign to the same variable. The program will remember the latest values assigned to it. Having the computer remember information, such as the amount of missed calls on your phone, and then planning to update or change the variable when you miss another call, is a significant component of programming.

- Rules for assigning names to variables: -
- Variable names should start with either letter or an underscore i.e., ("_"). The rest of the characters are not allowed at the start of a variable.
- Variable names should not be "keywords" of Python like "class." Such keyword can be ('while', 'if', 'else', 'lambda', 'is', 'return'). Python has around 30 keywords in total.
- Variable names should not contain special characters like "$".

Expressions

Values, variables, operators, and function calls, all come together to form an expression. The Python interpreter evaluates any expression entered at the prompt and presents the outcome. In the following example, Python uses "len" function which calculates the number of characters in a string while the second example shows just an addition of two integers.

```
>>> len('hello')
5

>>> 2 + 5
7
```

Expressions can appear on the right side of assignment statements because their evaluation results in a value. Both a value and a variable are straightforward expressions on their own.

In Python, a class, variable, or function is defined and identified by a name called an identifier. An operand is a thing that is being worked on. An operator, on the other hand, is a unique symbol that carries out the arithmetic or logical operations on the operands. The following table illustrates the operand and its meaning.

Expressions can be divided into the following types:

1 Constant Expressions: -
The values for the variable are constant until and unless the user changes it as per requirement.

```
>>> x = 10 + 15
>>> print(x)
25
```

2 Arithmetic Expressions: -
The components of an arithmetic expression include numerical numbers, operators, and occasionally parentheses. This kind of expression also yields a numeric value as a result. These expressions contain arithmetic operators, such as addition, subtraction, and others.

Operator	Syntax	Working
+	x + y	Addition or summation of x and y.
-	x – y	Subtraction of y from x.
*	x * y	Multiplication or product of x and y.
/	x / y	Division of x and y.
//	x // y	Quotient when x is divided by y.
%	x % y	Remainder when x is divided by y.
**	x ** y	Exponent (x to the power of y).

Table 1.1: Python Arithmetic Operators

3 Integral Expressions: -
These are the kinds of expressions that are produced in the form of integers only after all computations and type conversions. The example below will help to clarify.

```
>>> a = 5
>>> b = 7.1
>>> c = a + int(b)
>>> print(c)
12
```

4 Floating Expressions:
These are the kinds of expressions that produce floating point numbers as a result of all computations and type conversions.

```
>>> a = 5
>>> b = 7.1
>>> c = a + b
>>> print(c)
12.1
```

5 Arithmetic expressions: -
These are written on both sides of the relational operator (>„ >=, =) in expressions referred to as relational expressions. These mathematical expressions are first assessed, followed by a comparison using a relational operator, which results in a Boolean output. These are also known as Boolean expressions.

```
>>> a = 21
>>> b = 13
>>> c = 40
>>> d = 37

>>> >>> p = (a + b) >= (c - d)
>>> print(p)
True
```

6 Logical Expressions: -
Expressions that have a True or False outcome are known as logical expressions. In essence, it provides a requirement or conditions. For instance, the statement (10 == 9) is true if 10 equals 9. Since we are aware that it is incorrect, it will return False. When studying logical expressions, we also encounter various logical operators that are frequently used in logical expressions. Here are a few Python logical operators:

Operator	Syntax	Working
and	x and y	The expression returns True if both x and y are true,; it returns False

or	x or y	The expression returns True if at least one of *x* or *y* is True.
Not	not x	The expression returns True if the condition of x is False.

Table 1.2: Python Logical Operators

1.1.2 NUMERIC AND STRING OPERATIONS

Numeric Operations

These operations are performed generally on numbers. The operations yield either float or integer value depending on the operations. The standard operators used for arithmetic operations are addition, subtraction, multiplication, and so on. These operators are listed in the table given above. The use of these operators is easy and does not need a separate explanation.

String Operations

Python uses string operations to work with strings. They can be used to replace a character, find a substring, or change the case of a string. String formatting and word splitting are both possible using string operations. The intention behind these operations is, sometimes the required format of the string is different than the input data provided. So, the user needs to convert the input string into the required format and then perform further processing.

Function Name	Description
capitalize()	Converts the first character of the string to a capital (uppercase) letter
count()	Returns the number of occurrences of a substring in the string.
index()	Returns the position of the first occurrence of a substring in a string
isalnum()	Checks whether all the characters in a given string are alphanumeric or not

Function Name	Description
isalpha()	Returns “True” if all characters in the string are alphabets
isdecimal()	Returns true if all characters in a string are decimal
isdigit()	Returns “True” if all characters in the string are digits
is lower()	Check if all characters in the string are lowercase
isnumeric()	Returns “True” if all characters in the string are numeric characters
join()	Returns a concatenated String
lower()	Converts all uppercase characters in a string into lowercase
replace()	Replace all occurrences of a substring with another substring
startswith()	Returns “True” if a string starts with the given prefix
strip()	Returns the string with both leading and trailing characters
swapcase()	Converts all uppercase characters to lowercase and vice versa
title()	Convert string to title case
upper()	Converts all lowercase characters in a string into uppercase

Table 1.3: Python String Operation Examples

1.2 DATA STRUCTURES

What is Data Structure

The expression "Data + Structures" says it all. The data is organized by a data structure. It is a storage device that arranges and keeps data in a way that the programmer can quickly access. Data management, storage, and organization are crucial since they make it possible for efficient changes and easier access. With the use of data structures, you may set up your data in a way that makes it possible to store groups of data, link them together, and carry them out. operations on them, as necessary.

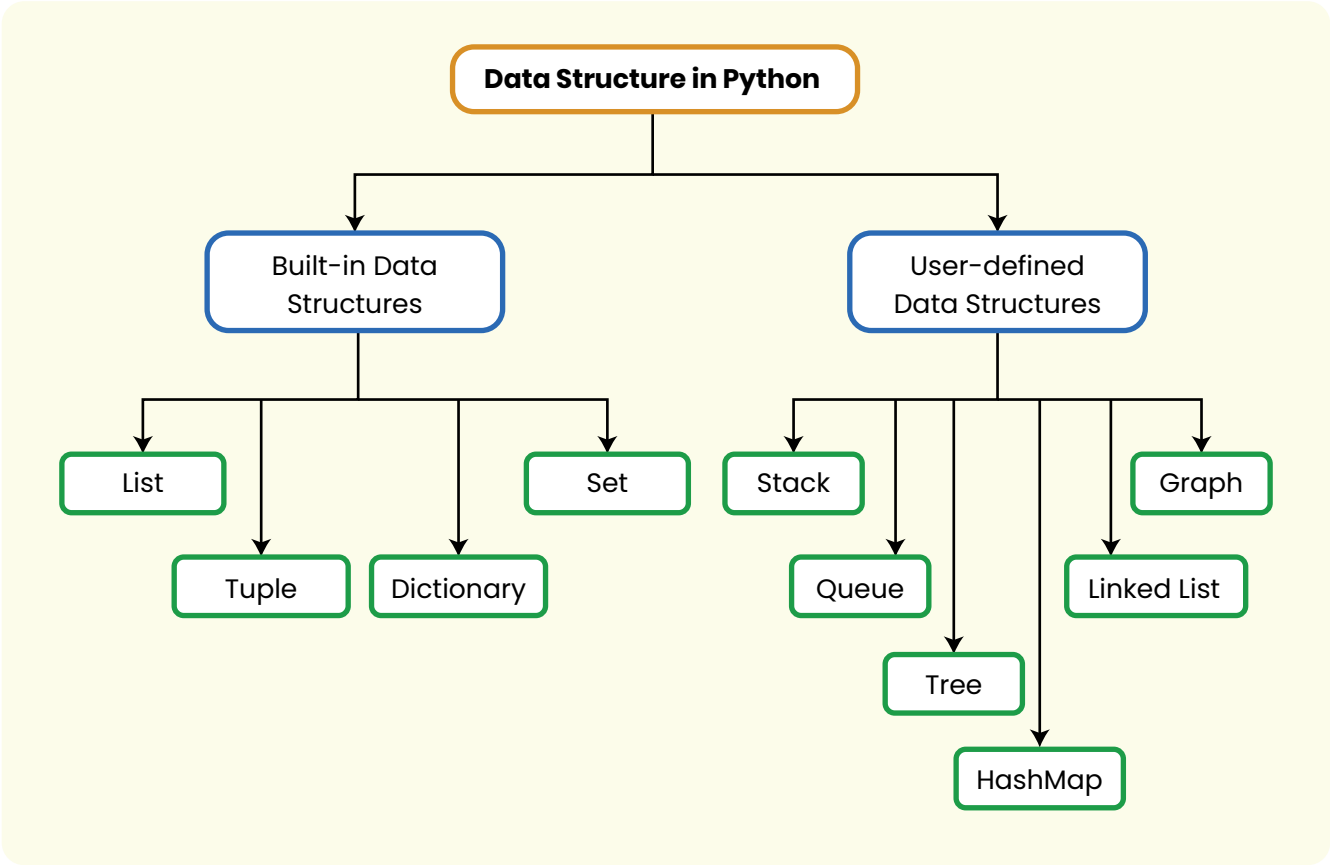


Fig1.1: Types of Data Structures in Python

1.2.1 LIST

In Python, the sequence of various data types is stored in a list. A list is a built-in data structure in Python, which is the collection of different kinds of values or items. Every item in the list, also known as the Index, has an address allocated to it. The index value begins at 0 and continues until the final component, which is the positive index. You can also retrieve elements from last to first using negative indexing, which starts at -1.

Creating a list

Lists are created using square brackets, and then the elements are added accordingly. If none of the elements are entered in the square brackets, then an empty list would be created.
Example:

```
>>> list1 = ['Pune', 'Mumbai', 123, 'Banglore']
>>> print(list1)
['Pune', 'Mumbai', 123, 'Banglore']
```

Characteristics of list

- Ordered: The items in the list are ordered, which implies they have a specific order which will not change on the addition of new elements, since the newly added elements would be placed at the end.
- Mutable: The items in the list are mutable, which implies that we can change, add, and remove items in a list after it has been created.
- Allow Duplicates: Lists allow duplicate values, since the items in the list are ordered and indexed.

  Example:

```
>>> list1 = ['Pune', 'Mumbai', 123, 'Banglore', 'Pune']
>>> print(list1)

['Pune', 'Mumbai', 123, 'Banglore', 'Pune']
```

Adding elements to list

Since lists are mutable this implies items can be added to the list. Elements can be added using append(), extend(), and insert() functions.

- Using append() function
  The append method allows us to add the elements passed to it as a single element at the end of the list.

  Example:

```
>>> list1 = ['Pune', 'Mumbai', 123, 'Banglore', 'Pune']
>>> list1.append(['Chennai',789])
>>> print(list1)

['Pune', 'Mumbai', 123, 'Banglore', 'Pune', ['Chennai', 789]]
```

- Using extend () function
  The extend method allows us to add the elements passed to it one by one at the end of the list.

  Example:

```
>>> list1 = ['Pune', 'Mumbai', 123, 'Banglore', 'Pune']
>>> list1.extend(['Chennai',789])
>>> print(list1)

['Pune', 'Mumbai', 123, 'Banglore', 'Pune', 'Chennai', 789]
```

- Using insert() function
  The insert method allows you to add the elements at the specified index to the list.

  Example:

```
>>> list1 = ['Pune', 'Mumbai', 123, 'Banglore', 'Pune']
>>> list1.insert(1,'Chennai')
>>> print(list1)

['Pune', 'Chennai', 'Mumbai', 123, 'Banglore', 'Pune']
```

Deleting items from the list

- Elements can be deleted from the list using,
- del statement: Removes an item from the list using its index.
- remove () method: The remove method eliminates the list's first instance of the provided value.
- pop () method: An item at a given index is removed from the list via the pop() method, and then it is returned. It eliminates and returns the last item if an index is not specified.

  Example:

```
>>> list1 = ['Pune', 'Mumbai', 123, 'Banglore', 'Pune']
>>> del list1[1]      # Using del keyword
>>> print(list1)
['Pune', 123, 'Banglore', 'Pune']

>>> list1.remove('Banglore')
>>> print(list1)      # Using remove method
['Pune', 123, 'Pune']

>>> list1.pop(2)
>>> print(list1)      # Using pop method
['Pune', 123]
```

Other Functions

Several other functions are available with lists, some of them are as follows

- len(): Calculates the length of the list by returning the number of elements present in it.
- index(): The index() method returns the index of the first instance of the given item in the list.
- sort(): The list's contents are sorted in ascending order using the sort() method. The reverse argument can be used to sort in descending order as well.
- reverse(): The reverse method allows us to reverse the order of the elements in the list.

1.2.2 TUPLES

A tuple is a built-in data structure in Python similar to lists but also has some significant differences. The difference between the two is that once the elements are assigned to the tuple can't be changed. Tuples are used to store an ordered series of elements and are defined by parentheses ().

Characteristics of tuples

- Immutable: Tuples are immutable, which means that once you construct a tuple, you cannot change its contents (add, remove, or modify elements) like you do with lists. On the other hand, you can build a new tuple using an old one.
- Ordered: Like lists, tuples keep the order of their elements. A tuple has an index for each element that starts at 0.
- Heterogeneous: Tuples can store elements from several data types i.e., they are heterogeneous.

Creating a tuple

Tuples can be created using the following methods.

- Using Parentheses '()': Parentheses around comma-separated values are the most common approach in creating a tuple. This is how a tuple frequently gets defined.

  Example:

```
>>> tuple1 = ('Pune', 'Mumbai', 123, 'Banglore', 'Pune')
>>> print(tuple1)

('Pune', 'Mumbai', 123, 'Banglore', 'Pune')
```

- Using the 'tuple()' constructor: The tuple() constructor that is already built in can also be used to produce tuples. The constructor will transform an iterable (like a list) into a tuple if you pass it an argument.

  Example:

```
>>> list1 = ['Pune', 'Mumbai', 123, 'Banglore', 'Pune']
>>> tuple1 = tuple(list1)
>>> print(tuple1)

('Pune', 'Mumbai', 123, 'Banglore', 'Pune')
```

Accessing elements

Since tuples are ordered collections, each element has a corresponding index. A tuple's elements can be accessed in the following ways:

- Indexing: By enclosing the index of the desired element in square brackets [] you can retrieve a specific element within a tuple, as the indexing begins at 0. To access elements from the end of the tuple, you can alternatively use negative indices. The last element is denoted by the number 1, the next-to-last by the number 2, and so on.

  Example:

```
>>> tuple1 = ('Pune', 'Mumbai', 123, 'Banglore', 'Pune')
>>> second_element = tuple1[1]
>>> last_element = tuple1[-1]
```

- Slicing: Slicing allows you to extract a range of elements from a tuple. The colon: operator found between square brackets is used for slicing. It gives back a fresh tuple with a set number of elements.

  Example:

```
>>> tuple1 = ('Pune', 'Mumbai','Banglore')
>>> for element in tuple1:
>>>     print(element)

Pune
Mumbai
Banglore
```

Other Functions

Several other functions are available with tuples, some of them are as follows.

- len(): The len() method returns the number of elements present in the tuple.
- index(): The index() method provides the first instance of a given value in the tuple's index.
- sorted(): The method returns a new, sorted list with each of the tuple's elements in it.
- min() and max(): The methods return the minimum and maximum value from the tuple respectively

Python Lists	**Python Tuples**
Lists are mutable.	Tuples are immutable.
Since lists are mutable, they have several built-in methods.	Due to their immutability, tuples have fewer built-in functions than other data types.
Iterations are time-consuming	Iterations are comparatively Faster
Python lists are created using square brackets '[]'	Python tuples are created using parentheses '()'

Table 1.5: Comparison between Lists and Tuples

1.3 CONDITIONAL STATEMENTS

Statements that have the structure "If A then B" belong to conditional statements. A is referred to as the hypothesis statement, whereas Q is called as the conclusion statement. Programming languages use conditional statements to instruct a computer on what action should be taken in response to specific situations. If and only if the previously stated conditions are true or false, these decisions are made.

Types of conditional statements in Python are:

if statement

If conditional statement allows a further block of the program to execute if and only if the given statement under the if constructor is True.

If (condition):

If the statement is useful if user must check against only one condition.

e.g. – if the user input number is greater than 10, then do further calculations.

```
>>> num=int(input('Enter the num '))
>>> if num>10:
        print('Do futher calculations')

Enter the num  11
Do futher calculations
```

if-else statement: -

It includes an if statement along with the else statement.

In an if-else statement, the else statement gets automatically executed if the if-statement turns out to be false.

The user need not separately mention the conditions for else statement.

Syntax for If -Else statement is.

If condition 1:

Statement 1

Else:

Statement 2

```
>>> num=int(input('Enter the num '))
>>> if num>10:
        print('Do futher calculations')
    else:
        print('U are not allowed for further calculations')

Enter the num  9
U are not allowed for further calculatiosn
```

if-elif-else statement: -

If user has to check for more than two conditions, then the if-elif-else statement comes into the picture.

If the if-condition is false, then it will check for the elif-condition and if the elif-condition is also false, then it will automatically execute the else condition.

Users can use multiple elif conditions between the if and else statements.

Syntax for if-elif-else statement:

If (condition 1):

Statement 1

elif (condition 2):

Statement 2

elif (condition 3):

statement 3

else:

Statement 4

```
>>> num=int(input('Enter the num '))
>>> if num>10:
    print('addition is allowed ')
>>> elif num < 10 and num >100:
    print('Multiplication is allowed')
>>> else:
    print('cannot do further calculations')

Enter the num  15
addition is allowed
```

Nested if – statements: -

Nested if structures can be made by combining many conditional statements within one another. It is useful when a user must check for many conditions.

Syntax for Nested if –statement:

If (condition 1):

If (condition 2):

else:

else:

```
>>> num=int(input('Enter the num '))
>>> if num>10:
>>>     if num<20:
            print('Addition is allowed')
        else:
            print('Multiplication is allowed')
    else:
        print('U are not allowed for further calculations')

Enter the num  15
Addition is allowed
```

Loops in Python: -

The loop is a powerful method in any programming language to automate tasks rather than doing things manually. Loops are very useful for repeating the block of code. For example, if one has to repeat the same message 50 times, rather than typing the same message 50 times, one can use a loop to do the same task just by typing the message one time only.

Two main types of loops in Python are:

- For Loop
- While Loop

For loop: -

For loop can be useful to iterate elements over the list, string, and range functions.

For loop over the list: -

loop runs for a number of times as that of the length of a specified list with each element in the list as loop input.

```
>>> lst=[1,2,3,4,5,6]
>>> for i in lst:
        print(i,end=' ')

1 2 3 4 5 6
```

For loop with Range() Function: For loop is widely used with range function which takes input from the user and repeat the block of code for a specified time.

Arguments in range function:

range(n): n is user-defined integer value and loop starts with n=0 and runs for (n-1) times.
e.g. range(5) will take values 0,1,2,3,4

range(a, b): loop will start running from a and will end at b-1
e.g. range(1,5) will take values 1,2,3,4

range(a, b, c) : Loop will run from a to b-1 with step size of c
e.g. range(1,10,2) will take values 1,3,5,7,9

```
>>> for i in range(1,10):
        print(i,end=' ')

1 2 3 4 5 6 7 8 9
```

For loop over string: -
For loop can be iterated over the string with each character as input to the loop.

```
>>> str1='Mango Apple'
>>> for i in str1:
        print(i ,end= ' ')

M a n g o   A p p l e
```

While loop: In Python, a while loop is a programming construct that enables us to repeatedly run a block of code as long as the specific condition is true. Each time the loop iterates, the condition is assessed. The block of code is run if the condition is satisfied. The loop ends if the condition is false.

```
>>> counter=0
>>> while counter<10:
        print(counter, end=' ')
        counter=counter+1


0 1 2 3 4 5 6 7 8 9
```

Here in the above example, the counter is set at 0 and after each loop 1 is added to counter while the counter <10. once counter value is set to 10, while condition turns out to be false, and loop stops.

1.4 FILE OPERATIONS

An important piece of data stored on a computer is a file. Each file can be recognized by its filename and file extension. Python is a very sophisticated programming language that is increasingly being used in a variety of different industries. One of its miracles is the way it handles and manages files, which would save us a ton of time.

A wide range of Python functions may handle the need for file operations like opening, reading, writing, creating files, etc. Before performing any file operations, a file must first be opened. For generating, reading, updating, and deleting files, Python includes several functions. The primary Python function for handling files is the open() function. The open() function takes two parameters; filename, and mode. There are four different methods (modes) for opening a file:

"r" - Read - Default value. Opens a file for reading, error if the file does not exist.
"a" - Append - Opens a file for appending and creates the file if it does not exist.
"w" - Write - Opens a file for writing and creates the file if it does not exist.
"x" - Create - Creates the specified file and returns an error if the file exists.

The sample code is shown below.

```
>>> File_name = open("filename.txt", ”r”)
```

The open() function returns a file object, which has a read() method for reading the content of the file:

```
>>> File_name = open("filename.txt", ”r”)
>>> print(File_name.read())
```

If the file is at a different location, you will have to specify the file path, like this:

```
>>> File_name = open("D:\\myfiles\ filename.txt", "r")
>>> print(File_name.read())
```

It is a good practice to always close the file when you are done with it.

```
>>> File_name = open("filename.txt", "r")
>>> print(File_name.readline())
>>> File_name.close()
```

To create a new file in Python, use the open() method, with one of the following parameters:
"x" - Create - will create a file, return an error if the file exists.
"a" - Append - will create a file if the specified file does not exist.
"w" - Write - will create a file if the specified file does not exist.

```
>>> File_name = open("filename.txt", "x")
```

This will create a new empty file.
To delete a file, you must import the OS module, and run its os.remove() function:

```
>>> import os
>>> os.remove("filename.txt")
```

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 Which of the following is the correct way to concatenate two lists, list1 and list2, in Python?
a) list1.append(list2)
b) list1.extend(list2)
c) list1 + list2
d) list1.merge(list2)

2 Which keyword is used to start a for loop in Python?
a) start
b) begin
c) for
d) loop

3 Which of the following is a valid way to assign the value 25 to a variable named x in Python?
a) x = 25
b) x == 25
c) 25 = x
d) x: = 25

4 Which mode is used to open a file for reading in Python?
a) r
b) w
c) a
d) x

5 What is the primary purpose of a for loop in Python?
a) To print the numbers from 1 to N.
b) To execute a block of code repeatedly until a condition is met.
c) To define a function in Python.
d) To iterate over a sequence and perform a specific action for each item.

Answers

1: (c) *2: (c)* *3: (a)* *4: (a)* *5: (d)*

DESCRIPTIVE QUESTIONS

1 Explain what a variable is in Python and why they are important in programming.

2 Define a Python list and explain its characteristics.

3 Explain the difference between append () and extend () methods for lists.

4 Explain the difference between for and while loops in Python.

5 Explain the purpose of file handling in Python and how you can open and close files

CHAPTER 2
DATA ANALYSIS FOR PYTHON

LEARNING OBJECTIVES

- To understand the importance of Python libraries in data analysis.
- Learn how to import and utilize external libraries in Python.
- Master NumPy's role in numerical computing and array manipulation.
- To understand pandas' importance for structured data manipulation and analysis.
- To understand the importance of data preprocessing in preparing data.
- Recognize EDA's role in data understanding and visualization.

2.1 INTRODUCTION TO LIBRARIES

Packages refers to a directory of sub-packages and modules, whereas a module is a file containing Python code. In other words, a library is a group of files, also referred to as modules, that include functions that other programs can utilize. In other words, Python Libraries are reusable code that we may incorporate into our programs. It has code bundles that can be utilized repeatedly in various programs. It makes Python programming simpler and easier for programmers. After that, we won't have to write the same code for other projects. Data visualization, machine learning, and computer science are just a few of the disciplines that significantly rely on Python libraries.

2.1.1 WORKING OF PYTHON LIBRARIES:

Understanding the workings of Python libraries like Pandas and NumPy is essential for effectively using them in data manipulation and analysis tasks. Let's take a closer look at how these libraries work:

- **NumPy:**
  NumPy, short for "Numerical Python," is a foundational library for numerical and scientific computing in the Python programming language. It is the go-to library for performing efficient numerical operations on large datasets, and it serves as the backbone for numerous other scientific and data-related libraries.
  a **Array Representation:**
  At the core of NumPy is the nd-array object, which is a multi-dimensional array. These arrays can hold elements of the same data type and are the building blocks of numerical computations. NumPy arrays are highly memory-efficient and optimized for performance.
  b **Data Storage:**
  NumPy stores data in a contiguous block of memory, allowing for efficient element-wise operations. It uses a C-style array internally, which means that elements are stored in a linear, one-dimensional sequence.

elements are stored in a linear, one-dimensional sequence.

c **Vectorized Operations:**
NumPy promotes vectorized operations, which means that mathematical operations are performed on entire arrays, rather than iterating through elements one by one. This vectorization leads to significant performance improvements.

d **Universal Functions (ufuncs):**
NumPy provides a wide range of universal functions (ufuncs) that operate elementwise on arrays. These ufuncs are implemented in compiled C code, making them extremely fast.

e **Broadcasting:**
NumPy enables broadcasting, allowing you to perform operations on arrays with different shapes. When arrays have different shapes, NumPy automatically expands the smaller one to match the larger one's shape.

f **Indexing and Slicing:**
NumPy arrays support advanced indexing and slicing, allowing you to access specific elements or subarrays efficiently.

g **Mathematical Functions:**
NumPy offers a comprehensive set of mathematical functions for array manipulation, including statistics, linear algebra, and more.

- **Pandas:**
  a **Data Structures:**
  Pandas primarily revolves around two core data structures: Series and DataFrame.
    - **Series:** A one-dimensional labeled array that can hold data of any type.
    - **DataFrame:** A two-dimensional labeled data structure, resembling a spreadsheet or SQL table, with rows and columns.

  b **Data Alignment:**
  Pandas is designed for data alignment. When you perform operations on Series or DataFrame, Pandas automatically aligns data based on index labels, making it easier to work with data even when they have different sizes or missing values.

  c **Label-Based Indexing:**
  Pandas emphasizes label-based indexing, which allows you to access and manipulate data based on the labels (row and column names) rather than numerical indices.

d **Data Cleaning:**
Pandas provides powerful tools for data cleaning and preparation, including handling missing data, changing data types, and applying functions to data.

e **Data Aggregation:**
You can easily aggregate data using Pandas by grouping rows based on some criteria and applying functions like sum, mean, or custom aggregation functions.

f **Data Merging and Joining:**
Pandas supports merging and joining Data-Frames based on common columns or indices, similar to SQL operations.

g **Data Visualization Integration:**
Pandas integrates well with data visualization libraries like Matplotlib and Seaborn, allowing you to create informative plots and graphs from your data. Pandas provides functions for reading data from various file formats (CSV, Excel, SQL databases) and writing data to these formats.

NumPy and Pandas are crucial libraries for data manipulation and analysis in Python. NumPy focuses on efficient numerical operations with arrays, while Pandas extends this functionality to structured data with labelled rows and columns, making it an ideal choice for data cleaning, exploration, and preparation. Understanding their underlying mechanisms helps you leverage their full potential in your data science projects.

Feature	Pandas	NumPy
Data Structures	Series, DataFrame	ndarray (NumPy arrays)
Data Types	Supports heterogeneous data types	Supports homogeneous data types
Indexing	Labeled indexing (rows and columns)	Integer-based indexing (arrays)
Missing Data Handling	Supports handling missing data	No built-in support for missing data
Data Grouping and Aggregation	Offers groupby and aggregation operations for summarizing data.	Typically used for numerical aggregation without labeled groups.
Data Visualization Integration	Works well with data visualization libraries like Matplotlib and Seaborn.	Typically used in conjunction with data visualization libraries for plotting.
Memory Efficiency	May use more memory, especially for large DataFrames due to data structure overhead.	Typically, more memory-efficient for numerical data due to homogeneous arrays.

Table 2.1: Comparison between Numpy and Pandas

2.1.2 FOUNDATIONAL OPERATIONS IN NUMPY AND PANDAS

A BASIC METHODS IN NUMPY

I Importing NumPy:

To use NumPy in Python, you first need to import it:

```
import numpy as np
```

The common convention is to alias NumPy as `np`.

II Creating Arrays:

NumPy arrays are the fundamental data structure. You can create arrays using various methods, such as:

```
# Creating a 1D array
>>>arr = np.array([1, 2, 3])
>>>arr
array([1, 2, 3])


# Creating a 2D array (matrix)
>>>matrix = np.array([[1, 2, 3], [4, 5, 6]])
>>>matrix
array([[1, 2, 3],
       [4, 5, 6]])


# Creating arrays of zeros or ones
>>>zeros = np.zeros((3, 3))
array([[0., 0., 0.],
       [0., 0., 0.],
       [0., 0., 0.]])

>>>ones = np.ones((2, 2))
array([[1., 1.],
       [1., 1.]])

# Creating a range of values
>>>rng = np.arange(0, 10, 2)  # Start at 0, stop before 10, step by 2
array([0, 2, 4, 6, 8])
```

III Basic Operations:

NumPy allows you to perform element-wise operations on arrays. For example:

```
a = np.array([1, 2, 3])
b = np.array([4, 5, 6])

# Addition
>>>Addition = a + b
array([5, 7, 9])

# Subtraction
>>>Subtraction = a - b
array([-3, -3, -3])

# Multiplication
>>>Multiplication = a * b
array([ 4, 10, 18])

# Division
>>>Division = a / b
array([0.25, 0.4 , 0.5 ])
```

IV Array Shape and Dimensions:

Check the shape and dimensions of an array using the `shape` and `ndim` attributes:

```
>>>arr = np.array([[1, 2, 3], [4, 5, 6]])
>>>print(arr.shape)  # (2, 3) - 2 rows, 3 columns
(2, 3)

>>>print(arr.ndim)   # 2 - 2 dimensions (2D array)
2
```

V Indexing and Slicing:

NumPy supports indexing and slicing to access elements or subsets of arrays. Indexing starts at:

```
my_list = [1, 2, 3, 4, 5]

# Indexing: Accessing individual elements
>>>first_element = my_list[0]  # Access the first element (1)
>>>first_element
1
```

```
# Negative indexing: Counting from the end
>>>last_element = my_list[-1]   # Access the last element (5)
>>>last_element
5

# Slicing: Extracting a portion of the list
>>>sub_list = my_list[1:4]  # Extract elements from index 1 to 3 (2, 3, 4)
>>>sub_list
[2, 3, 4]

# Slicing with step: Extract elements with a step of 2
>>>step_list = my_list[0:5:2]  # Extract elements with a step of 2 (1, 3, 5)
>>>step_list
[1, 3, 5]

# Slicing from the beginning or to the end
>>>partial_list = my_list[:3]  # Extract elements from the start to index 2 (1, 2, 3)
>>>partial_list
[1, 2, 3]

>>>partial_list_end = my_list[2:]  # Extract elements from index 2 to the end (3, 4, 5)
>>>partial_list_end
[3, 4, 5]
```

VI Aggregation and Statistics:

NumPy provides functions for computing various statistics on arrays

i Aggregation:

```
# Example dataset: A list of exam scores
exam_scores = [85, 92, 88, 78, 95, 89, 92, 87, 91, 84]

# Aggregation: Calculating summary statistics
>>>total_scores = sum(exam_scores)  # Calculate the sum of all scores
>>>total_scores
881

>>>average_score = total_scores / len(exam_scores)  # Calculate the mean (average) score
>>>average_score
88.1

>>>min_score = min(exam_scores)  # Find the minimum score
>>>min_score
78

>>>max_score = max(exam_scores)  # Find the maximum score
>>>max_score
95

>>>median_score = sorted(exam_scores)[len(exam_scores) // 2]  # Calculate the median score
>>>median_score
88.5
```

ii Statistics:

```
# Statistics: Measures of central tendency and dispersion
import statistics

>>>mean_score = statistics.mean(exam_scores)  # Mean using the statistics module
>>>mean_score
88.1

>>>median_score = statistics.median(exam_scores)  # Median using the statistics module
>>>median_score
88.5

>>>variance = statistics.variance(exam_scores)  # Variance using the statistics module
>>>variance
24.1

>>>std_deviation = statistics.stdev(exam_scores)  # Standard deviation using the statistics module
>>>std_deviation
4.909175083453431
```

VII Reshaping and Transposing:

Reshaping and transposing are fundamental operations when working with multi-dimensional data, such as matrices or arrays. These operations allow you to change the structure or dimensions of your data.

i **Reshaping:** Reshaping involves changing the shape or dimensions of your data while maintaining the total number of elements. This operation is often used in machine learning and data preprocessing to prepare data for modeling.

ii **Transposing:** Transposing involves switching the rows and columns of a two-dimensional data structure like a matrix or array. This operation is particularly useful for linear algebra operations or when working with tabular data.

```
# Example dataset: A 2D array (matrix)
matrix = np.array([[1, 2, 3],
                   [4, 5, 6],
                   [7, 8, 9]])

# Reshaping: Changing the shape of the matrix
>>>reshaped_matrix = matrix.reshape((3, 3))  # Reshape the matrix into a 3x3 matrix
>>>reshaped_matrix
array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9]])

#flattened_matrix
>>>flattened_matrix = matrix.flatten()  # Flatten the matrix into a 1D array
>>>flattened_matrix
array([1, 2, 3, 4, 5, 6, 7, 8, 9])
```

```
# Transposing: Switching rows and columns
>>>transposed_matrix = np.transpose(matrix)  # Transpose the matrix
>>>transposed_matrix
array([[1, 4, 7],
       [2, 5, 8],
       [3, 6, 9]])

# Alternatively, you can use matrix.T to transpose it
```

VIII Universal Functions (ufuncs):

NumPy provides universal functions that operate element-wise on arrays, including trigonometric, logarithmic, and exponential functions.

```
>>>arr = np.array([1, 2, 3])
>>>exp_arr = np.exp(arr)     # Exponential function
>>>exp_arr
array([ 2.71828183,  7.3890561 , 20.08553692])

>>>sin_arr = np.sin(arr)     # Sine function
>>>sin_arr
array([0.84147098, 0.90929743, 0.14112001])

>>>log_arr = np.log(arr)     # Natural logarithm
>>>log_arr
array([0.        , 0.69314718, 1.09861229])
```

IX Random Number Generation:

NumPy includes functions for generating random numbers from various distributions, such as `np. random. rand`, `np. random. rand`, and `np. random. rand`.

```
>>>random_array = np.random.rand(3, 3)  # Generate a 3x3 array of random values between 0 and 1
>>>random_array

array([[0.45169709, 0.35940579, 0.43168611],
       [0.07886612, 0.55081402, 0.95071264],
       [0.03285086, 0.72657747, 0.36759608]])
```

X Broadcasting:

NumPy allows you to perform operations on arrays of different shapes, often automatically aligning their shapes, thanks to broadcasting rules.

```
>>>arr = np.array([1, 2, 3])
>>>result = arr + 5  # Broadcasting: [1, 2, 3] + [5, 5, 5] = [6, 7, 8]
array([6, 7, 8])
```

XI Reshaping Arrays:

reshape arrays into different dimensions using np. reshape or the reshape method.

```
>>>arr = np.array([1, 2, 3, 4, 5, 6])
>>>reshaped = np.reshape(arr, (2, 3))  # Reshape into a 2x3 matrix
array([[1, 2, 3],
       [4, 5, 6]])
```

B BASIC METHODS IN PANDAS:

I Importing Pandas

To harness the power of Pandas, start by importing the library with the common alias pd. You'll include this line at the beginning of your Python script or Jupiter Notebook:

```
import pandas as pd
```

II Creating a DataFrame

The core data structure in Pandas is the DataFrame, which is a two-dimensional, labeled data structure with columns of potentially different types.

i Creating a DataFrame from a Dictionary:

DataFrames are at the heart of Pandas, allowing you to structure and work with data seam

```
# Create a DataFrame from a dictionary
>>>data = {'Name': ['Alice', 'Bob', 'Charlie'],
        'Age': [25, 30, 35]}
>>>df = pd.DataFrame(data)

    Name    Age
0   Alice   25
1   Bob     30
2   Charlie 35
```

ii Creating a DataFrame from External Data Sources:

Pandas provides functions for reading data from external sources, such as CSV files, Excel spreadsheets

```
df = pd.read_csv('data.csv')
```

III Viewing Data

Once you have a DataFrame, you can quickly examine its contents using several methods:

- **df. head ():** Displays the first few rows of the DataFrame.
- **df.tail():** Displays the last few rows of the DataFrame.
- **df.shape:** Returns the dimensions of the DataFrame (number of rows, number of columns).
- **df.columns:** Returns the column labels.

```
# Display the first few rows
print(df.head())

# Display the last few rows
print(df.tail())

# Get the dimensions of the DataFrame
print(df.shape)

# List the column labels
print(df.columns)
```

These methods are invaluable for getting an initial sense of your data's structure and content.

IV Indexing and Selecting Data

The process of choosing certain subsets of your data during data analysis is one of the most frequent activities. Pandas offers a variety of techniques for choosing and indexing data from a DataFrame.

```
# Selecting a single column
name_column = df['Name']

# Selecting multiple columns
subset = df[['Name', 'Age']]

# Filtering data based on a condition
young_people = df[df['Age'] < 30]
```

V Sorting Data

Sorting data helps in organizing it for better analysis or presentation. You can sort a DataFrame based on one or more columns.methods:

```
# Sorting by a single column in ascending order
df.sort_values(by='Age', ascending=True, inplace=True)

# Sorting by multiple columns
df.sort_values(by=['Age', 'Name'], ascending=[True, False], inplace=True)
```

VI Data Aggregation and Summary Statistics

Pandas provides functions for computing various summary statistics on your data, such as mean, sum, median, standard deviation, and more. You can also group your data based on one or more columns and then apply aggregation functions.

```
# Computing the mean of a column
mean_age = df['Age'].mean()

# Grouping data by a column and computing statistics
grouped_data = df.groupby('Category')['Value'].sum()
```

VII Adding and Dropping Columns

You can add new columns to your DataFrame or drop existing ones as needed. Adding columns allows you to create calculated fields, while dropping columns helps you remove unnecessary data.

```
# Adding a new calculated column
df['New_Column'] = df['Column1'] + df['Column2']

# Dropping a column
df.drop(columns=['Column_to_Drop'], inplace=True)
```

VIII Handling Missing Data

Real-world data often contains missing values. Pandas provides methods for detecting and dealing with missing data, such as filling missing values with a specific value or removing rows or columns containing missing data.

```
# Checking for missing values
missing_values = df.isna().sum()

# Filling missing values
df['Column'].fillna(value, inplace=True)

# Dropping rows with missing values
df.dropna(subset=['Column_with_missing_values'], inplace=True)
```

IX Merging and Concatenating DataFrames

In many cases, you'll need to combine data from multiple sources. Pandas offers functions for merging and concatenating DataFrames based on common column or indices.

```
# Merging two DataFrames based on a common column
merged_df = pd.merge(df1, df2, on='Common_Column')

# Concatenating DataFrames vertically
concatenated_df = pd.concat([df1, df2], axis=0)
```

X Saving and Loading Data

After processing and analyzing data, you may want to save your results or load data for further analysis. Pandas provides methods to save and load DataFrames in various formats, including CSV, Excel.

```
# Saving a DataFrame to a CSV file
df.to_csv('output_data.csv', index=False)

# Loading data from an Excel file
loaded_df = pd.read_excel('input_data.xlsx')
```

2.2 DATA PREPROCESSING:

Data preparation is an essential stage in the pipeline for data analysis and machine learning. To prepare raw data for analysis or the training of machine learning models, it must be cleaned, transformed, and organised. Here are some typical data pretreatment steps:

IMPORTANCE OF DATA PREPROCESSING:

- **Data Quality Improvement:** Real-world data is often messy, containing missing values, duplicates, outliers, and errors. Data preprocessing helps identify and address these issues, leading to higher data quality and more reliable results.

- **Enhanced Model Performance:** Clean and well-pre-processed data can lead to better model performance. Machine learning models often struggle with noisy or incomplete data, and preprocessing helps mitigate these issues.

- **Extraction and Engineering:** Data preprocessing allows you to create new features or transform existing ones to better represent underlying patterns in the data. This can improve a model's ability to capture complex relationships.

- **Normalization and Scaling:** Scaling features to a similar range (e.g., standardization or normalization) can help models converge faster and avoid certain optimization problems in algorithms like gradient descent.

- **Handling Categorical Data:** Many machine learning algorithms require numerical input, so preprocessing techniques like one-hot encoding or label encoding are essential for converting categorical data into a format suitable for modeling.

- **Dimensionality Reduction:** In high-dimensional datasets, preprocessing techniques like dimensionality reduction (e.g., PCA) can help reduce computational complexity and remove noise from the data.

- **Improved Interpretability:** Cleaned and well-pre-processed data is easier to interpret and understand. It simplifies the analysis process and allows for more meaningful insights.

DATA PREPROCESSING STEPS:

- **Data Collection:** Gather the raw data from various sources, such as databases, files, APIs, or sensors.

- **Data Cleaning:**
  - **Handling Missing Values:** Identify and handle missing data, which can involve filling in missing values with default values, using interpolation, or removing rows/columns with missing data.

```
#Filling missing values with default values:
import pandas as pd

# Load your dataset
data = pd.read_csv("your_dataset.csv")

# Fill missing values with a default value, e.g., 0
data.fillna(0, inplace=True)

# Use linear interpolation to fill missing values
data.interpolate(method='linear', inplace=True)

# Remove rows with any missing values
data.dropna(inplace=True)
```

  - **Removing Duplicates:** Identify and remove duplicate records if they exist in the dataset.

```
# Remove duplicate records based on all columns
data.drop_duplicates(inplace=True)
```

- **Outlier Detection and Treatment:** Detect and handle outliers that may skew the analysis or model training. This can involve removing, transforming, or imputing outlier values.

```
# create the data
data = np.array([1, 2, 3, 4, 5, 10, 20, 30])
# assign your quartiles, limits and iq3
q1, q3 = np.percentile(data, [25, 75])
iqr = q3 - q1
lower_bound = q1 - 1.5*iqr
upper_bound = q3 + 1.5*iqr
#create conditions to isolate the outliers
outliers = data[(data < lower_bound) | (data > upper_bound)]
print(outliers)
```

DATA TRANSFORMATION: Data transformation is a critical step in the data preprocessing pipeline, which involves converting data into a suitable format for analysis, modeling, or visualization. It helps improve the quality of data, making it more understandable and valuable for various data-related tasks. Here are some specific data transformation techniques and their explanations:

- **Normalization/Standardization:**
  - **Normalization:** Scaling all numerical features to a range between 0 and 1. This is useful when the features have different scales.

  - **Standardization:** Transforming data to have a mean of 0 and a standard deviation of 1. It assumes a normal distribution of data.

```
from sklearn.preprocessing import MinMaxScaler, StandardScaler

# Normalize data to [0, 1]
min_max_scaler = MinMaxScaler()
normalized_data = min_max_scaler.fit_transform(data)

# Standardize data (mean=0, std=1)
std_scaler = StandardScaler()
standardized_data = std_scaler.fit_transform(data)
```

- **Log Transformation:** Applying the natural logarithm (or other logarithmic functions) to data to reduce skewness in distributions and make them more symmetric.

```
import numpy as np

# Log transformation
log_transformed_data = np.log(data)
```

- **Encoding Categorical Variables:** Converting categorical variables into numerical format, often using techniques like one-hot encoding or label encoding.more symmetric.

```
from sklearn.preprocessing import OneHotEncoder, LabelEncoder

# One-hot encoding
one_hot_encoder = OneHotEncoder()
encoded_data = one_hot_encoder.fit_transform(categorical_data)

# Label encoding
label_encoder = LabelEncoder()
encoded_labels = label_encoder.fit_transform(categorical_labels)
```

- **Binning/Discretization:** Grouping continuous data into discrete bins or categories, which can help simplify complex datasets.

```
from sklearn.preprocessing import KBinsDiscretizer

# Binning into discrete intervals
binning = KBinsDiscretizer(n_bins=3, encode='ordinal', strategy='uniform')
binned_data = binning.fit_transform(data)
```

- **Feature Engineering:** Creating new features from existing ones to capture important information or patterns in the data.

```
# Example: Creating a feature for age groups

data['age_group'] = pd.cut(data['age'], bins=[0, 18, 35, 60, np.inf], labels=['child', 'young', 'adult', 'senior'])
```

- **Aggregation and Grouping:** Aggregating data by grouping rows based on certain attributes and applying summary statistics.

```
# Grouping by a categorical variable and calculating mean

grouped_data = data.groupby('category')['value'].mean()
```

DATA REDUCTION:

- **Dimensionality Reduction:** If dealing with high-dimensional data, reduce the number of features using techniques like Principal Component Analysis (PCA) or feature selection methods.

- **Principal Component Analysis (PCA):** PCA is a popular technique for reducing the dimensionality of a dataset while preserving the most important information. It does this by identifying the principal components, which are linear combinations of the original features that capture the most variance in the data.

```
from sklearn.decomposition import PCA

pca = PCA(n_components=2)  # You can specify the number of components to keep
reduced_data = pca.fit_transform(data)
```

  After applying PCA, you retain a reduced set of features (principal components) that explain most of the variance in the data. This can be particularly useful for visualization or as a preprocessing step before applying machine learning algorithms.

- **Feature Selection:**
  Feature selection methods aim to select the most informative features from the original set while discarding less important ones. Common techniques include:
  Univariate Feature Selection: Selecting features based on statistical tests like chi-squared or mutual information.

```
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2

selector = SelectKBest(score_func=chi2, k=5)  # Select the top 5 features
selected_features = selector.fit_transform(data, target)
```

- **Recursive Feature Elimination (RFE):** Iteratively removing the least important features based on a model's performance.

```
from sklearn.feature_selection import RFE
from sklearn.linear_model import LogisticRegression

model = LogisticRegression()
rfe = RFE(model, n_features_to_select=5)  # Select the top 5 features
selected_features = rfe.fit_transform(data, target)
```

  Feature selection can improve model performance, reduce overfitting, and speed up training by focusing on the most relevant features.

DATA IMBALANCE HANDLING

Handling data imbalance is essential when dealing with datasets where one class significantly outnumbers another. This imbalance can lead to biased model training and reduced predictive performance. Here's a concise overview of common techniques for handling data imbalance:

- **Oversampling:**
  - Oversampling the minority class by creating additional copies of its instances.
  - Techniques include Random Oversampling, SMOTE (Synthetic Minority Over-sampling Technique), and ADASYN (Adaptive Synthetic Sampling).
  - It increases the number of minority class samples, making the class distribution more balanced.

```
from imblearn.over_sampling import RandomOverSampler

oversampler = RandomOverSampler(random_state=42)
X_resampled, y_resampled = oversampler.fit_resample(X, y)
```

- **Undersampling:**
  - Undersampling the majority class by randomly removing some of its instances.
  - It reduces the number of majority class samples to match the minority class.

```
from imblearn.under_sampling import RandomUnderSampler

undersampler = RandomUnderSampler(random_state=42)
X_resampled, y_resampled = undersampler.fit_resample(X, y)
```

- **Synthetic Data Generation (SMOTE):**
  - SMOTE creates synthetic examples in the minority class by interpolating between existing samples.
  - It helps diversify the dataset and balance the class distribution.

```
from imblearn.over_sampling import SMOTE

smote = SMOTE(random_state=42)
X_resampled, y_resampled = smote.fit_resample(X, y)
```

2.2.1 EXPLORATORY DATA ANALYSIS (EDA)

Exploratory data analysis (EDA) is the process of analysing and examining record sets in order to understand their main characteristics, find patterns, find outliers, and find correlations between variables. EDA is typically performed as a first step before to more formal statistical studies or modelling.

- **Fundamental Objectives of EDA:**
  - **Understand the Data:** EDA's primary objective is to thoroughly comprehend the dataset you're dealing with. Understanding the structure, variables, and connections between the data is required for this. It involves acquiring a sense of what the data means and the questions it might be able to answer.
  - **Detect Patterns and Trends:** EDA aims to uncover patterns, trends, and relationships within the data. This can include identifying correlations between variables, seasonality in time series data, or clusters in data points.
  - **Identify Anomalies and Outliers:** EDA helps in the early detection of anomalies and outliers in the data. These unusual data points can be errors or carry valuable information. Identifying them is crucial for data quality and model performance.
  - **Assess Data Quality:** EDA involves evaluating the quality of the data, including missing values, inconsistencies, and data entry errors. Ensuring data quality is essential for reliable analysis and modeling.
  - **Formulate Hypotheses:** EDA often leads to the generation of hypotheses or research questions. By exploring the data visually and statistically, you can develop hypotheses about relationships between variables or potential insights to investigate further.
  - **Select Relevant Features:** During EDA, you can determine which features (variables) are most relevant to the analysis or modeling task. This helps in feature selection and dimensionality reduction.
  - **Guide Data Preprocessing:** EDA informs the data preprocessing steps needed before analysis or modeling. It helps decide how to handle missing data, outliers, and feature transformations.
  - **Visualization and Communication:** EDA often involves creating visualizations to present data insights effectively. Clear visualizations can convey complex information to both technical and non-technical audiences.

- **Drive Decision-Making:** The insights gained through EDA can guide decision-making processes. Whether it's in business, science, or any other field, EDA provides a data-driven foundation for making informed decisions.

- **Types of EDA:**

  EDA (exploratory data analysis) is a term that refers to a variety of methods and strategies for drawing conclusions from data. While there aren't any clear-cut "types" of EDA, there are a few standard techniques and approaches that data analysts may use to explore and comprehend their datasets. Following are a few EDA's main strategies.

  - **Univariate Analysis:** This sort of evaluation makes a speciality of analyzing character variables inside the records set. It involves summarizing and visualizing a unmarried variable at a time to understand its distribution, relevant tendency, unfold, and different applicable records. Techniques like histograms, field plots, bar charts, and precis information are generally used in univariate analysis.

  - **Bivariate Analysis:** In a bivariate evaluation, the relationships between the variables are examined. It makes it possible to identify relationships, dependencies, and correlations between different pairs of data. The most common techniques in bivariate analysis include scatter plots, line plots, correlation matrices, and move-tabulation.

  - **Multivariate Analysis:** Multivariate analysis extends bivariate evaluation to encompass greater than variables. It ambitions to apprehend the complex interactions and dependencies among more than one variable in a records set. Techniques inclusive of heatmaps, parallel coordinates, aspect analysis, and primary component analysis (PCA) are used for multivariate analysis.

  - **Time Series Analysis:** Statistical sets with a temporal component are the major subjects of this kind of study. In order to evaluate a time collection, styles, features, and seasonality within the data across the years are examined and modelled. In time series analysis, methods including line graphs, autocorrelation analysis, transferring averages, and ARIMA (Autoregressive Integrated Moving Average) patterns are often used.

  - **Missing Data Analysis:** Missing information is a not unusual issue in datasets, and it may impact the reliability and validity of the evaluation. Missing statistics analysis includes figuring out missing values, know-how the patterns of missingness, and using suitable techniques to deal with missing data. Techniques along with lacking facts styles, imputation strategies, and sensitivity evaluation are employed in lacking facts evaluation.

- **Outlier Analysis:** Statistics variables known as outliers significantly depart from the overall sample of the facts. Finding outliers and understanding their presence, their reasons for being, and their effects on the study are all part of outlier analysis. Outlier assessment methods include box plots, scatter plots, z-rankings, and clustering algorithms.

- **Data Visualization:** Data visualisation, a crucial component of EDA, involves producing visual representations of the data to aid with comprehension and investigation. Different visualisation methods, including as bar charts, histograms, scatter plots, line plots, heatmaps, and interactive dashboards, are used to depict different types of statistics.

These are just a few examples of the types of EDA techniques that can be employed at some stage in information evaluation. The choice of strategies relies upon on the information traits, research questions, and the insights sought from the analysis.

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 What does NumPy stand for?
a) Numerical Python
b) Numeric Processing
c) Number Processing
d) Numeric Manipulation

2 Which data structure is fundamental to NumPy for handling numerical data efficiently?
a) Lists
b) Tuples
c) Arrays
d) Dictionaries

3 Which statistical method is commonly used to detect outliers in a numeric dataset?
a) Standardization
b) Interquartile Range (IQR)
c) Mode calculation
d) Z-Score calculation

4 Which data balancing technique aims to increase the size of the minority class by generating synthetic data points?
a) Oversampling
b) Under sampling
c) Feature engineering
d) Principal Component Analysis (PCA)

5 What is the main purpose of exploratory data analysis (EDA)?
a) To make predictions about future data.
b) To uncover insights and patterns in data.
c) To preprocess data for machine learning models.
d) To create visualizations for presentations.

Answers

1: (a) *2: (c)* *3: (b)* *4: (a)* *5: (b)*

DESCRIPTIVE QUESTIONS

1 What is NumPy, and why is it essential for data science?

2 Comparing NumPy to Python lists, how does it improve array operations?

3 Explain the main objectives of exploratory data analysis (EDA).

4 What is the main advantage of using NumPy and Pandas for data manipulation?

5 Describe the sampling techniques used while dealing with imbalanced dataset.

CHAPTER 3
INTRODUCTION TO MACHINE LEARNING

LEARNING OBJECTIVES

- What is machine learning?
- Data preparation for building machine learning algorithms.
- Building, validating, and hyper-parameter tuning of ML algorithms.
- Basics of time series forecasting.

3.1 What comes to your mind after hearing Machine learning and AI? Is it robot, automatic machines?

But, machine learning is learning from historical data, understanding different patterns from data and making future predictions. Machine learning is a subset of Artificial Intelligence. The rapidly growing discipline of data science includes machine learning as a key element. Algorithms are trained using statistical techniques to make classifications or predictions and to find important insights from data. The decisions made because of these insights influence key growth metrics in applications and businesses, ideally. They will be expected to assist in determining the most pertinent business questions and the information needed to address them. In today's world, Machine Learning is getting widely used in many domains I.e., from Agriculture, Educations to Finance. From Self driving car, catboats to differentiating cancer cells from good cells. From Manufacturing to Maintenances industry. If AI is a vehicle, then Machine Learning is an engine to the vehicles.

Example of Machine Learning in different sectors:

- **Finance Industry:**
  Algo-Trading: To create trading strategies, ML algorithms can examine market data, news sentiment, and previous trading patterns.
  Credit Risk Analyzing: By analyzing the historical data of individual user, ML models can predict whether to credit the user or not and how much to credit.
  Fraud-Transaction detection: ML algorithms detect whether the transaction being done is normal or fraud and if it found fraud transaction, systems are there in place which send alert to customer.

- **HealthCare Industry:**
  Monitoring a patient remotely: Monitoring a patient remotely: Using cameras, other sensing devices, patients data Is getting captured and analyze continuously and if any anomaly is detected, alerts are send to respective doctor, till the doctors come, robots using ML can take preliminary's actions.
  ML in Radiology: On an average , radiologists can analyze 50 to 100 x-rays manually which might vary person to person but similar task can be achieve using Computer Vision (Subset of ML) with better accuracy. It can analyze millions of x-rays in few minutes.

- **Agriculture Industry:**
  Agri-Bot can be created using ML, NLP, and CV on village level which can sense the soil's parameter and can suggest the crop accordingly. It can understand local language and will be suggesting the farmer's different pesticides, different sprays in respective language just by looking at the defective leaf. Also it can forecast temperature, rain and other required parameters by analyzing the village level data.

- **Manufacturing and Maintenance Industry:**
  Manufacturing process flaws and anomalies are found using machine learning.
  Raw material requirement, raw material price, sales of products can be forecasted using ML.
  Predictive maintains is one of the great uses of ML.

These are just few applications of ML and now will see how to prepare data for ML algorithms.

3.2 TYPES OF ML ALGORITHMS

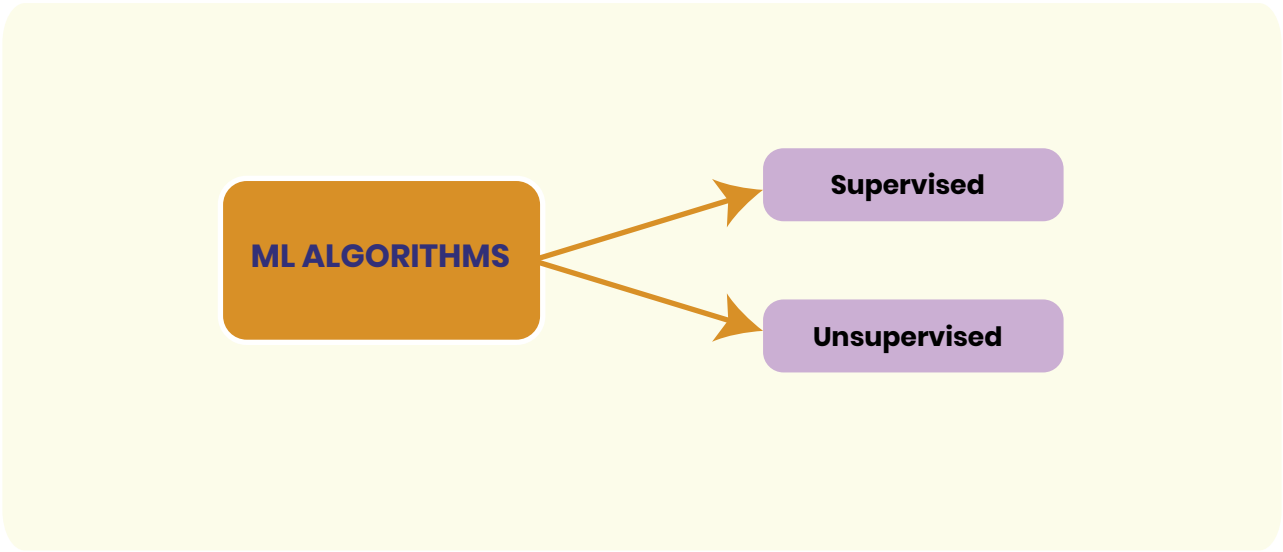
Machine learning based problem statements are classified as supervised, unsupervised and reinforcement learning algorithms.

Supervised Algorithm: The model is trained on a labeled dataset, which indicates that the input data is paired with corresponding output labels or target values. This is typically the case with supervised machine learning algorithms, a subset of machine learning techniques. To enable the model to correctly predict or classify newly discovered data, supervised learning aims to learn a mapping or relationship between the input features and the target variable.

Unsupervised Algorithm: Without labeled output or target values, unsupervised machine learning algorithms are used to discover patterns, relationships, or structures within data. For tasks like data exploration, dimensionality reduction, clustering, and density estimation, these algorithms are helpful.

Reinforcement learning: A type of machine learning model known as reinforcement learning (RL) teaches an agent how to make decisions by interacting with the environment around it. RL is based on trial-and-error learning, as opposed to supervised learning, where the algorithm is trained on labeled data, and unsupervised learning, where it indicates patterns in unlabeled data. By acting in the environment, the agent aims to maximize a cumulative reward signal.

In this module we are going to focus on supervised and un-supervised learning models.



CHAPTER 4
DATA VISUALIZATION WITH PYTHON

LEARNING OBJECTIVES

- Understanding different libraries used for Data Visualization in python.
- Learn about different types of plots and their use.
- Use of Matplotlib, Seaborn, and Plotly for Data visualization

The technique of illustrating data in a graphical style to make it easier to understand is commonly referred to as data visualization with Python. It is an effective method that can be utilized to convey data insights to a wide variety of audiences. Matplotlib, Seaborn, and Plotly are just a few of the several Python packages that can be used for data visualization. It is crucial to select the appropriate library for the work at hand since every library has different strengths and weaknesses.

After deciding on a library, you can begin generating data visualizations. The most widespread types of data visualizations include pie charts, scatter plots, bar charts, and line charts. Nevertheless, depending on the data you are using, there are a wide variety of additional visualizations that may be made. It is vital to adhere to some fundamental rules while creating data visualizations. Make sure your visualizations are first and foremost simple to interpret. This calls for utilizing unambiguous labeling, minimizing clutter, and selecting the best kind of display for your data.

Second, you should convey insights from your data through data visualizations. In other words, use colors, shapes, and sizes that will make it easier for your audience to understand the point you are making.

Finally, you should constantly check the accuracy of your data visualizations. This entails carefully reviewing your data and applying the appropriate statistical techniques.
Python is a strong tool that can be used to visualize data and convey insights to a range of audiences. You can make data visualizations that are both useful and educational by adhering to the standards.

4.1 BASIC PLOTS: LINE PLOT, SCATTER PLOT, HISTOGRAM

Line Plot

The graph used to show continuous data points on a number line is called a line plot. Plotting data points on the Cartesian plane first, then connecting those points with a number line, creates line graphs. Data points can be displayed via line plots for both single-variable and multiple-variable analysis. Line plots are mostly used when we have dates on the x-axis of the line plot. For Example,

```
# Plotting function y = x^2

import matplotlib.pyplot as plt

x = [1,2,3,4,5,6,7,8,9]
y = [1,4,9,16,25,36,49,64,81]

plt.xlabel('X')
plt.ylabel('Y')
plt.title('X vs Y')

plt.plot(x,y)
```

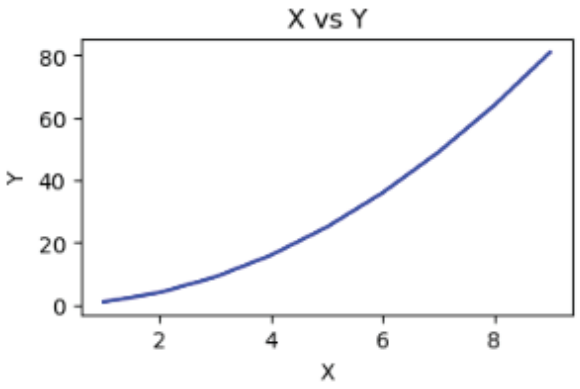


Fig. 4.1 Line Plot using Matplotlib

Just like the matplotlib, we can use Seaborn to generate the same plot. The only difference is that Seaborn does not accept the raw data. The data must come from a data frame. Then the syntax for plotting is very simple which is shown in the figure below.

```
# Plotting function y = x^2
import seaborn as sns

x = np.array([1,2,3,4,5,6,7,8,9])
y = np.array([1,4,9,16,25,36,49,64,81])
df = pd.DataFrame({'X':x,'Y':y})
sns.lineplot(x='X',y='Y',data=df)
```

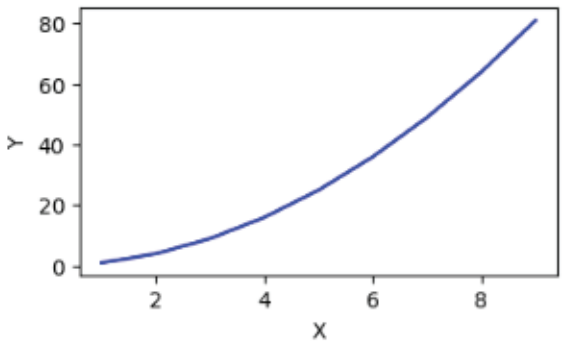


Fig. 4.2 Line Plot using Seaborn

Scatter Plot

These are the graphs/charts that employ Cartesian coordinates to observe and show relationships between variables. The variables' values (x for the first variable and y for the second) are shown as dots. The terms scatter plots, scattergrams, scatter graphs, scatter charts, and scatter diagrams are also used to describe them. It works well in circumstances when the dependent variable may take on different values depending on the independent variable.

This plot is mainly used for showing the relation between numeric variables like age v/s salary, height v/s weight, and so on. The scatter plot is also used to determine whether there is a positive or negative correlation between two variables. Positive correlation means one variable is increasing while the other one is also increasing while negative correlation is like one variable is increasing as a result of a decrease in other variable. For example,

```
# Plotting function y = x^2

import matplotlib.pyplot as plt

x = [1,2,3,4,5,6,7,8,9]
y = [1,4,9,16,25,36,49,64,81]

plt.scatter(x,y)
plt.xlabel('X')
plt.ylabel('Y')
plt.title('X vs Y')
```

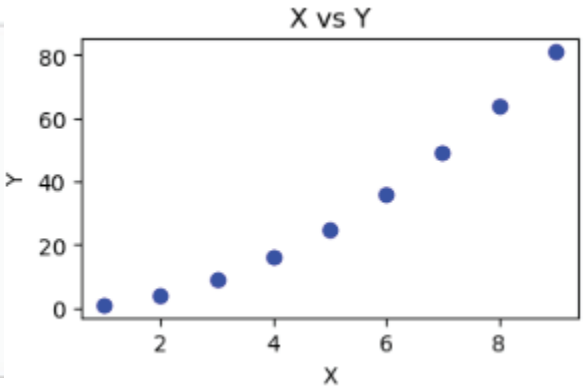


Fig. 4.3 Scatter Plot using Matplotlib

A scatter plot can also be generated by using Seaborn library. The syntax for seaborn with the resultant plot is shown in the figure below.

```
# Plotting function y = x^2
import seaborn as sns

x = np.array([1,2,3,4,5,6,7,8,9])
y = np.array([1,4,9,16,25,36,49,64,81])
df = pd.DataFrame({'X':x,'Y':y})
sns.scatterplot(x='X',y='Y',data=df)
```

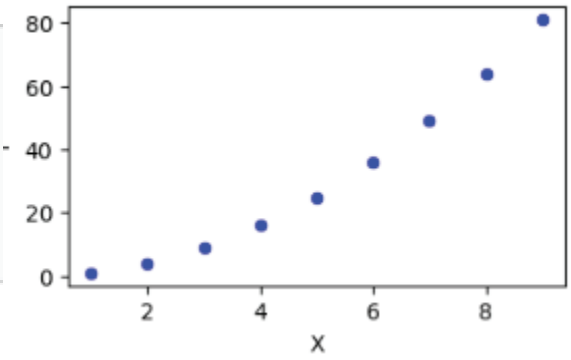


Fig. 4.4 Scatter Plot using Seaborn

Histograms

To visualize the frequency distribution of continuous variables, histograms are utilized. The histogram's length and width stand for frequency and interval, respectively. You must divide the interval into non-overlapping bins to construct a histogram. The use of a histogram makes it possible to examine data to find outliers and skewness. Also, it is used for visual verification of whether the data is normally distributed or not. Bin size plays an important role in the interpretation of histograms. If the bin size is too small, all the bins will have somewhat the same frequency while if the bin size is too large, all the values will be included in only a few bins which will not result in any useful conclusion. For example,

```
# Plotting function y = x^2

import matplotlib.pyplot as plt

y = [1,5,6,9,15,25,36,95,86,84,25,35,68,
     65,41,75,89,94,55,96,85,35,25,14,53,97,62,35]

plt.hist(y,color='blue')
```

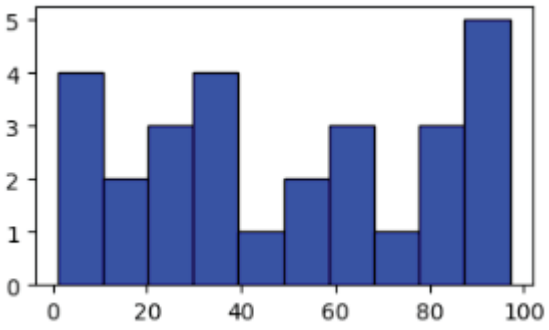


Fig. 4.5 Histogram Plot using Matplotlib

Seaborn uses the following syntax to plot histograms.

```
# Plotting function y = x^2

import seaborn as sns

y = [1,5,6,9,15,25,36,95,86,84,25,35,68,
     65,41,75,89,94,55,96,85,35,25,14,53,97,62,35]

sns.histplot(y,bins=10,color='green')
```

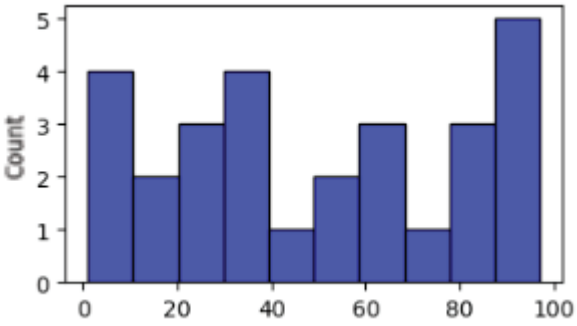


Fig. 4.6 Histogram Plot using Seaborn

Bar Plot

The bar plots are rectangular vertical and horizontal graphs that analyse data and enable you to see how things have changed over time as depicted by another axis, usually the X-axis. The value of one or more data points divided into a ratio can be stored in each bar. The more time a bar has, the more value it has. We express it using the bar() or barh() function in Matplotlib.

```
import matplotlib.pyplot as plt

plt.figure(figsize=(6,3.5))
plt.bar([0.25,2.25,3.25,5.25,7.25],[300,400,200,600,700],
label="Carpenter",color='b',width=0.5)
plt.bar([0.75,1.75,2.75,3.75,4.75],[50,30,20,50,60],
label="Plumber", color='g',width=.5)
plt.legend()
plt.xlabel('Days')
plt.ylabel('Wage')
plt.title('Details')
```

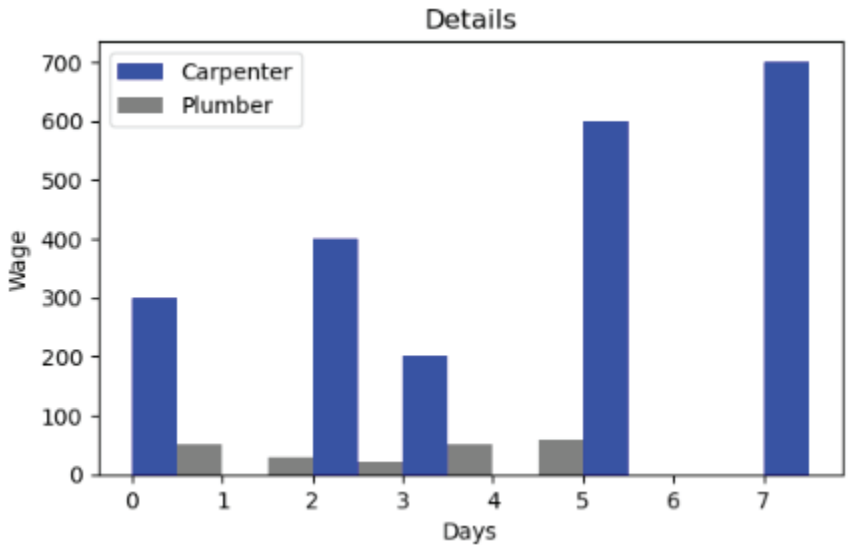


Fig. 4.7 Bar Plot using Matplotlib

As told earlier, for seaborn, it is better to arrange data in the data frame. Thus, to generate the same bar plot in seaborn, first data frame is created and then the bar plot is generated. The syntax for generating bar plot is shown below.

```
df1 = pd.DataFrame()
df1['Days'] = [0.25,2.25,3.25,5.25,7.25]
df1['No. of Workers'] = [300,400,200,600,700]
df1['Type of Workers'] = 'Carpenter'

df2 = pd.DataFrame()
df2['Days'] = [0.75,1.75,2.75,3.75,4.75]
df2['No. of Workers'] = [50,30,20,50,60]
df2['Type of Workers'] = 'Plumber'


df3 = pd.concat([df1,df2])
plt.figure(figsize=(4,2.25))
sns.barplot(x='Days',y='No. of Workers',data=df3,hue='Type of Workers',palette='Blues')
```

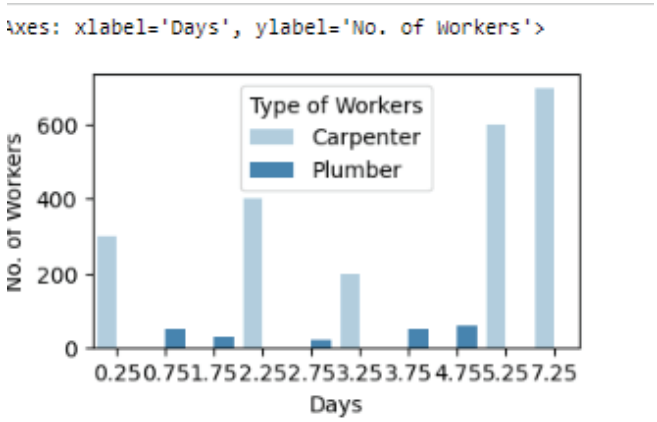


Fig. 4.8 Bar Plot using Seaborn

Bar plots are mainly used to represent a categorical variable against the numeric variable. It is also used to compare more than 1 categorical variable against the same numerical variable. Bar plots can also be stacked in nature. That is sub-categories of the main category will be accommodated one bar separated by color.

Pie Chart

A pie plot is a circular graph in which the data appears as slices, components, or parts of the pie. They serve the purpose of data analysts to display percentages or proportional statistics, with each pie slice denoting a different item or category of data. The pie() function in Matplotlib represents it.

```
1  import matplotlib.pyplot as plt
2
3  slice = [12, 25, 50, 36, 19]
4  activities = ['NLP','Neural Network', 'Data analytics',
5                'Quantum Computing', 'Machine Learning']
6  # cols = ['r','b','c','g', 'orange']
7
8  plt.figure(figsize=(6,3.5))
9  palette_color = sns.color_palette('Blues')
10 plt.pie(slice,labels =activities,
11 colors = palette_color,startangle = 90,autopct ='%1.1f%%')
12 plt.title('Training Subjects')
```

```
Text(0.5, 1.0, 'Training Subjects')
```

Fig. 4.9 Pie Chart showing Subject-wise Percentage Users

Pie charts are very convenient when values in terms of percentage are to be shown. But when the number of categories are more, the pie chart will create chaos, and visual will not be easy to interpret.

Boxplot

Boxplots are used to gauge how evenly dispersed a data set's data are. It makes three quartiles out of the data set. The minimum, maximum, median, first quartile, and third quartile of the data set are represented in this graph. Creating boxplots for each data set is also beneficial for comparing the distribution of data across data sets. To comprehend your data's distribution, whether it is skewed or not, and whether any outliers are present, boxplots are a useful tool for the visualization of data.

A boxplot is based on five important numbers: the "minimum," the "first quarter," the median, the "third quarter," and the "maximum." According to these definitions, the lowest and maximum values are Q1-1.5 * IQR and Q3 + 1.5 * IQR, respectively. Outliers are any points that are outside of these parameters.

Boxplots can be used to:

- Identify outliers or anomalous data points.
- To determine if our data is skewed.
- To understand the spread/range of the data

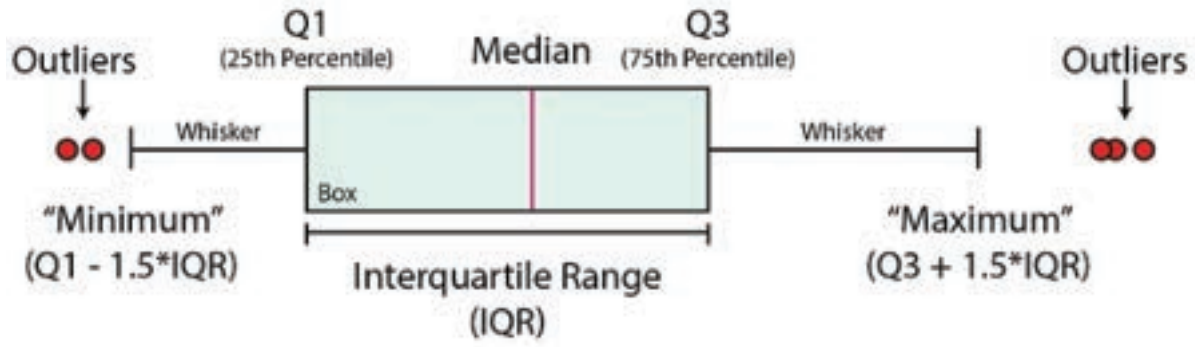


Fig. 4.10 Schematic of Box-Plot

```
1 import matplotlib.pyplot as plt
2 import numpy as np
3 np.random.seed(10)
4
5 data = np.random.normal(100, 20, 200)
6
7 plt.figure(figsize =(6,3.5))
8 plt.boxplot(data,vert=False)
```

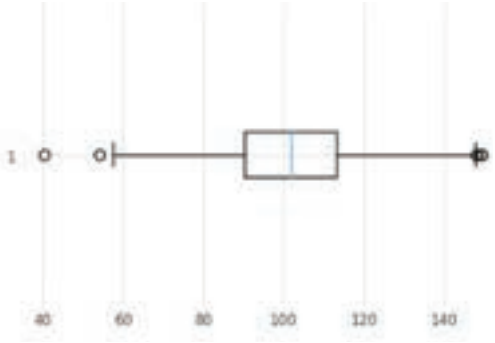


Fig. 4.11 Box-Plot using Matplotlib

Seaborn can generate boxplot in the following way,

```
import seaborn as sns
import numpy as np
np.random.seed(10)

data = np.random.normal(100, 20, 200)

plt.figure(figsize =(4,2.5))
sns.boxplot(data,orient='hori')
```

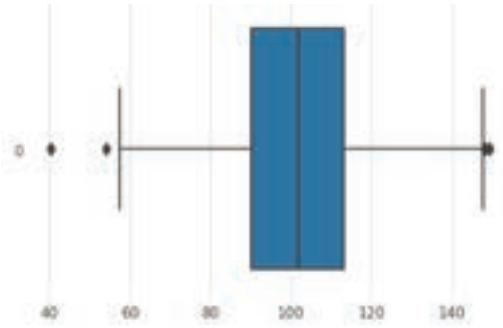


Fig. 4.12 Box-Plot using Seaborn

Heatmap

A heatmap is a graphic representation of data in which each value in a matrix is represented with a color. The individual values contained within a matrix are represented as colors in this two-dimensional graphical representation of the data. Because it can depict the relationship between variables, including time, it is an excellent method for visualizing data. the quantity of flights fluctuates over time, for instance.

The Seaborn package can be used to generate a heatmap in Python. A Python module called Seaborn is used to create statistical visuals. It is based on Matplotlib and offers a high-level interface for producing visually appealing and educational statistics visuals. The seaborn.heatmap() function can be used to generate a heatmap using Seaborn. A heatmap of the data is produced using the seaborn.heatmap() function, which accepts a 2D dataset as input. The link between the variables in the dataset is depicted by the heatmap, which is a matrix of colors.

The figure below shows the average temperature for each month in the last four years. Value for mean temperature can be seen directly from the plot. Heatmap is mainly used to plot the correlation between the numeric variables. This plot will help to determine which variables are positively correlated and which are negatively corelated.`

```
Temperature = np.array([15,26,32,35,45,12,19,24,26,39,37,29,
                        15 ,16, 17, 18, 19, 20, 20, 20, 20, 20,
                       25, 30, 35, 40, 45, 50, 50, 50, 50, 50,
                       30, 35, 40, 45, 50, 55, 60, 65, 70, 75,
                          35,40,25,28,32,18]).reshape(4,12)
months = ["January", 'February', 'March', 'April', 'May', 'June', 'July',
          'August', 'September', 'October', 'November', 'December']

year = [2019,2020,2021,2022]

df5 = pd.DataFrame(data=Temperature,columns=months,index=year)

plt.figure(figsize=(8,3))
sns.heatmap(df5,annot=True,cmap='crest')
```

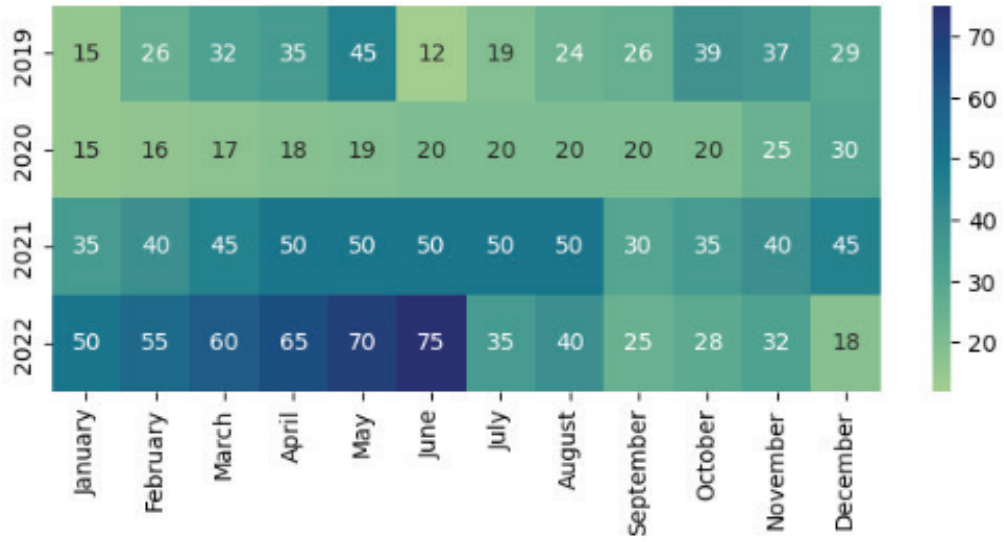


Fig. 4.13 Heatmap showing Mean Temperature for each Month.

Pair plot

Pairwise relationships in a dataset can be visualized using the pairplot() function in the Seaborn library of Python. Each variable in the data is shared over the y-axis across a single row, and the x-axis across a single column, thanks to the creation of an axis grid. A univariate distribution plot is created to display the marginal distribution of the data in each column for the diagonal plots, which are handled differently.

```
1  import seaborn
2  import matplotlib.pyplot as plt
3
4  df = seaborn.load_dataset('tips')
5
6  plt.figure(figsize=(5,3.5))
7  seaborn.pairplot(df, hue ='sex')
```

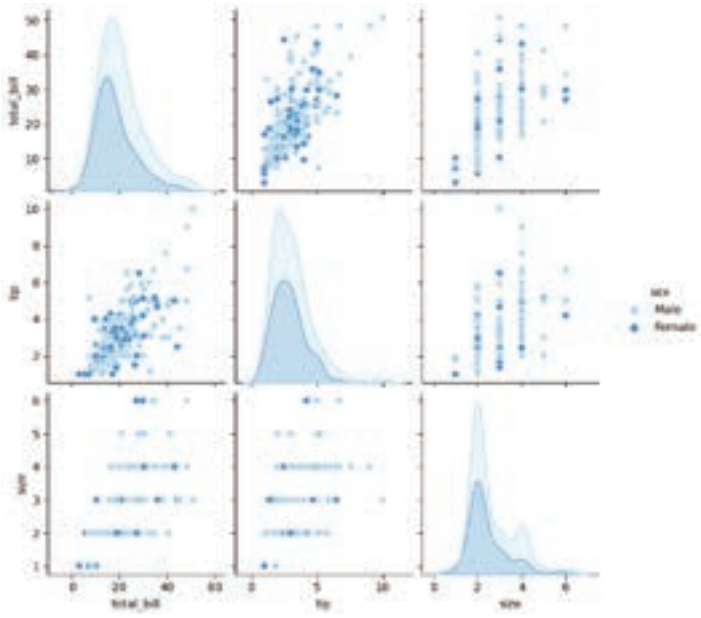


Fig. 4.14 Pair-Plot for Tips Dataset

Count Plot

The Python function countplot can be used to generate a bar chart that illustrates the counts of observations in each category bin. It may be used to find the most prevalent categories swiftly and effectively in a dataset, making it a very helpful function for data analysis. You must import the Seaborn library before calling the countplot method with the dataset and category variable.

Bar charts can be generated via countplot, which has a wide range of applications. For instance, a countplot can be used to make a bar chart that displays the number of observations in each group of a nested variable or a stacked bar chart that displays the number of observations in each category of two categorical variables. The most frequent categories in a dataset can be quickly and readily identified by employing the powerful data analysis tool countplot.

The example below shows number of people in dataset considered for plotting as per gender wise.

```
1 import seaborn as sns
2 import matplotlib.pyplot as plt
3
4 plt.figure(figsize=(4,2.5))
5 df = sns.load_dataset('tips')
6 sns.countplot(x ='sex', data = df,palette='Blues')
```

```
<Axes: xlabel='sex', ylabel='count'>
```

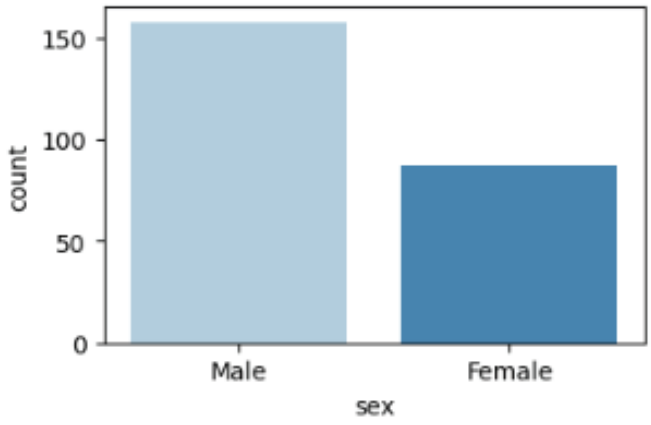


Fig. 4.15 Count-Plot showing Gender-wise Person count

Sunburst

A sunburst chart is a sort of visualization that Python programmers use to show hierarchical data. It resembles a treemap, but instead of being arranged in straight lines, it is organized in concentric rings, with the hierarchy's root at the center and its leaves at the edges. Sunburst diagrams are frequently employed to display the make-up of a whole, such as the distribution of costs in a budget or the age distribution of a population.

Sunburst graphs can be an effective tool for representing hierarchical data. They can be used to display a lot of information in a condensed amount of area and are simple to interpret. Consider utilizing a sunburst chart if you need to illustrate hierarchical data.

```
import plotly.express as px

df = px.data.tips()

px.sunburst(df, path=['day', 'sex'],
                  values='total_bill')
```

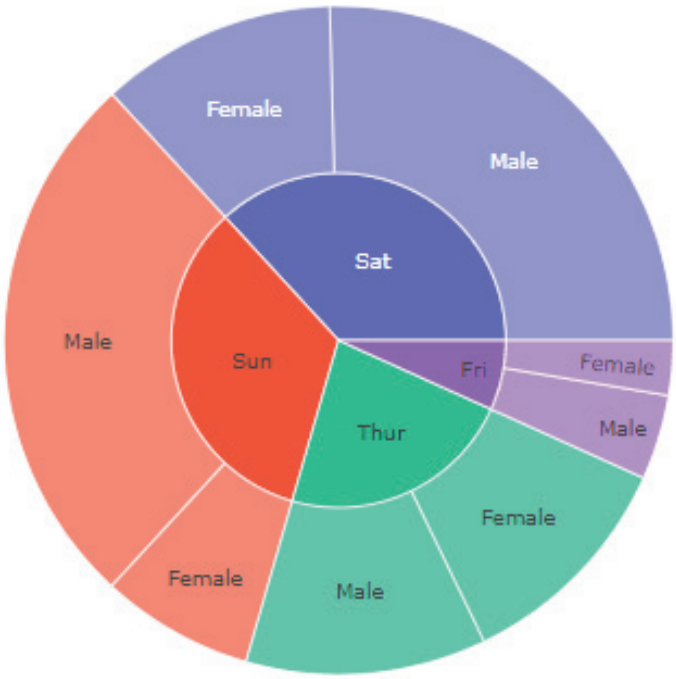


Fig. 4.16 Sunburst Plot showing Total Bill paid Day-wise and Gender-wise

Waterfall Chart

Waterfall plots (or charts) are widely employed to show how a specific value has changed cumulatively over time. In place of time, they can also employ pre-set categories (such as specific occurrences). As a result, this type of plot can be especially useful when giving presentations to business stakeholders because it makes it simple to demonstrate, for instance, how our company's income or client base has changed over time.

```
import plotly.graph_objects as go

fig = go.Figure(go.Waterfall(
    x = [["2016", "2017", "2017", "2017", "2017", "2018", "2018", "2018", "2018"],
       ["initial", "q1", "q2", "q3", "total", "q1", "q2", "q3", "total"]],
    measure = ["absolute", "relative", "relative", "relative", "total", "relative", "relative", "relative", "total"],
    y = [10, 20, 30, -10, None, 10, 20, -40, None], base = 300,
    decreasing = {"marker":{"color":"red", "line":{"color":"red", "width":2}}},
    increasing = {"marker":{"color":"green"}},
    totals = {"marker":{"color":"deep sky blue", "line":{"color":"blue", "width":3}}}
))

fig.update_layout(title = "Profit and loss statement", waterfallgap = 0.3)

fig.show()
```

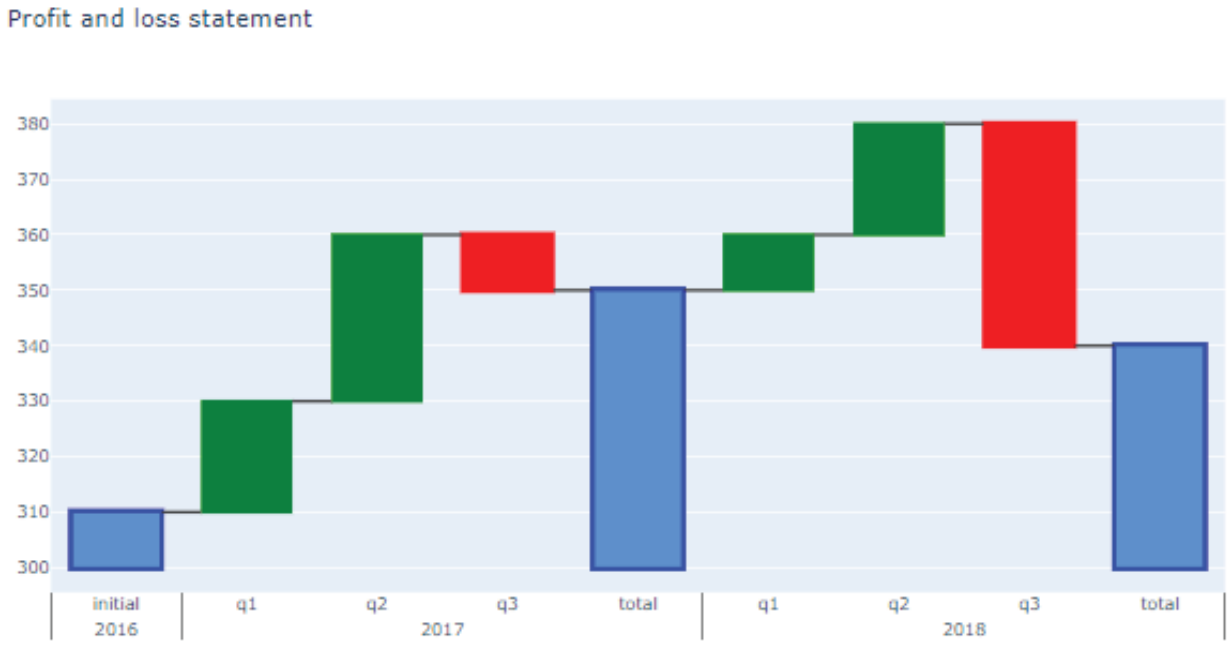


Fig. 4.17 Waterfall Chart showing Flow of Revenue

Donut Chart

Pie charts are transformed into donut charts by removing the central portion. In contrast to a pie chart, which focuses more on comparing the proportional area between the slices, a donut is more concerned with the usage of area of arcs to portray the information in the most effective way. Donut charts are more space-efficient because they can display more information about themselves in the blank spaces inside of them. It must be a Pie chart in order to be a Donut chart. If we examine the pie chart, we will pay particular attention to the chart's center. Contrarily, donut charts eliminate the need to compare the dimensions or areas of the slices and instead place the emphasis on the length of the arc, which is simple to calculate.

```
import matplotlib.pyplot as plt

Employee = ['Roshni', 'Shyam', 'Priyanshi',
'Harshit', 'Anmol']

Salary = [40000, 50000, 70000, 54000, 44000]

colors = ['#9999FF', '#0000FF', '#7777FF',
          '#5555FF', '#2222FF']
explode = (0.05, 0.05, 0.05, 0.05, 0.05)

plt.pie(Salary, colors=colors, labels=Employee,
        autopct='%1.1f%%', pctdistance=0.85,explode=explode)

centre_circle = plt.Circle((0, 0), 0.70, fc='white')
fig = plt.gcf()

fig.gca().add_artist(centre_circle)

```

```
<matplotlib.patches.Circle at 0x1a726c5f210>
```

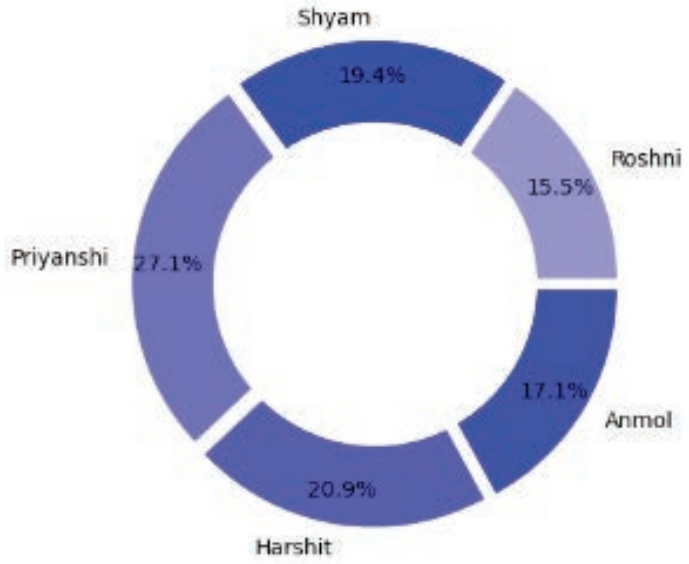


Fig. 4.17 Donut Chart Employees Salary Information

Dual Axis Chart

The most popular data visualization library, Matplotlib, allows us to create any type of plot; we can create a plot that has two y-axes and can provide different labels to both of the y-axes. We can make a plot with two different y-axes by creating or using two different axes items with the help of twinx() function. Sometimes, when analyzing any data through graphs, we need two x or y-axis to get some more insights into the data. This creates a concise plot that includes maximum information in one graph. Also if the user wants to compare trends of two variables in one graph, a dual axis serves that purpose.

```python
import pandas as pd
import matplotlib.pyplot as plt

dataset = pd.DataFrame({'Name':['Rohit', 'Seema',
'Meena', 'Geeta','Rajat'],'Height': [155,129,138,164,145],
'Weight': [60,40,45,55,60]})

ax = dataset.plot(kind = 'line', x = 'Name',
y = 'Height', color = 'Blue',linewidth = 3)

ax2 = dataset.plot(kind = 'line', x = 'Name',
            y = 'Weight', secondary_y = True,
            color = 'Red', linewidth = 3,
            ax = ax)

plt.title("Student Data")

ax.set_xlabel('Name', color = 'g')
ax.set_ylabel('Height', color = "b")
ax2.set_ylabel('Weight', color = 'r')

plt.tight_layout()
```

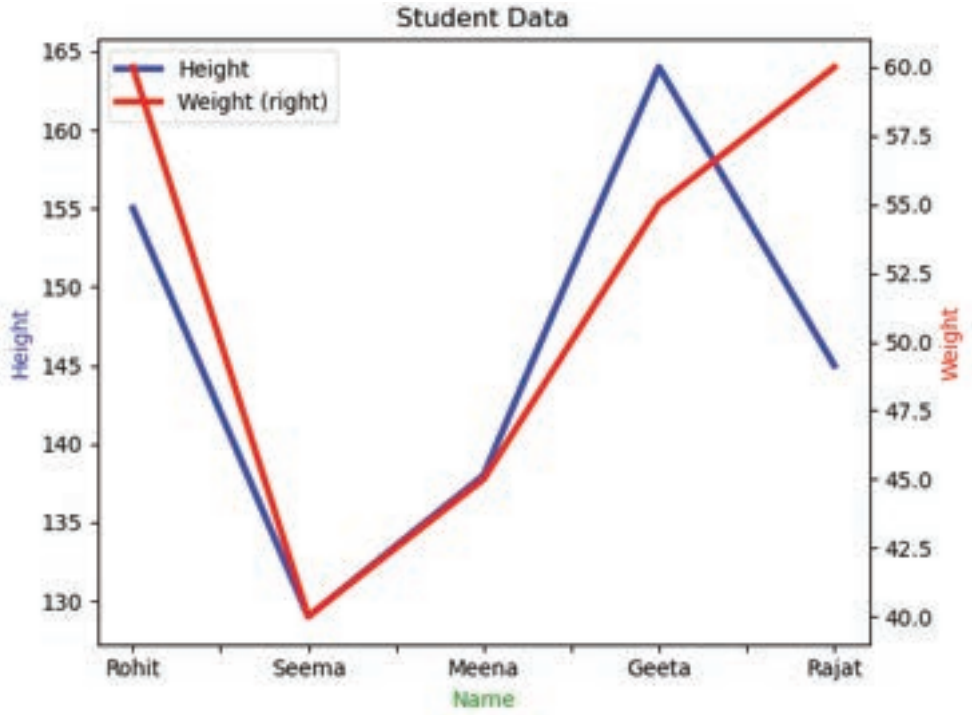


Fig. 4.18 Dual Axis Chart Person vs Height and Weight

Pareto Chart

A Pareto chart is a graphical data representation that emphasizes the key variables influencing a certain outcome. It is employed to illustrate the relative significance of many factors in a certain outcome and is named after economist Vilfredo Pareto of the 19th century. A bar graph and a line graph are both present in the graph, with the bars indicating the frequency of each factor and the line the cumulative total. One may arrive at better decisions based on data and concentrate efforts on the main contributing variables by examining the Pareto chart.

The ability to pinpoint the "vital few" elements that dominate the outcome is one of the main advantages of using a Pareto chart. Due to the ability to focus their efforts and resources on the areas that will have the biggest impact, organizations can do so. A Pareto chart, for example, might assist in determining the main causes of complaints, such as protracted wait times or defective products, if a business is experiencing a high number of client complaints. These crucial issues can be resolved by the business, which will increase consumer loyalty and happiness.

```
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.ticker import PercentFormatter

df = pd.DataFrame({'country': [177.0, 7.0, 4.0, 2.0,
                               2.0, 1.0, 1.0, 1.0]})
df.index = ['USA', 'Canada', 'Russia', 'UK',
            'Belgium', 'Mexico', 'Germany', 'Denmark']
df = df.sort_values(by='country',ascending=False)
df["cumpercentage"] = df["country"].cumsum()/df["country"].sum()*100


fig, ax = plt.subplots()
ax.bar(df.index, df["country"], color="blue")
ax2 = ax.twinx()
ax2.plot(df.index, df["cumpercentage"], color="red", marker="*")
ax2.yaxis.set_major_formatter(PercentFormatter())

ax.tick_params(axis="y", colors="blue")
ax2.tick_params(axis="y", colors="red")
```

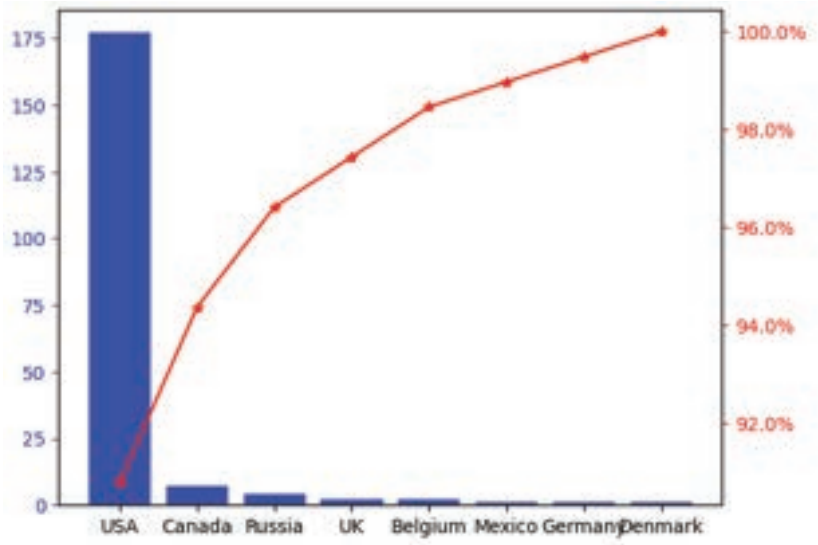


Fig. 4.19 Pareto Chart

Time series Plots

Time series analysis uses plots like Seasonal Decompose, Autocorrelation Function Plots (ACF Plots), and Partial Autocorrelation Function Plots (PACF Plots). These plots are explained one by one below.

Seasonal Decompose Plot

Decomposing a series into its level, trend, seasonality, and noise components is known as time series decomposition. A helpful abstract paradigm for thinking about time series generally and for better comprehending issues that arise during time series analysis and forecasting is decomposition. The division of a time series into systematic and non-systematic components is a useful abstraction for choosing forecasting techniques.

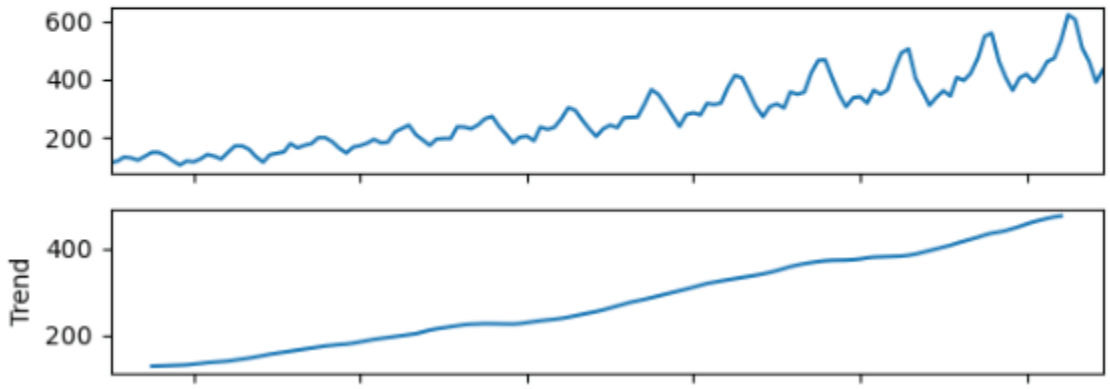
- **Systematic:** Components of the time series that have consistency or recurrence and can be described and modeled.
- **Non-Systematic:** Components of the time series that cannot be directly modeled.

A given time series is thought to consist of three systematic components including level, trend, seasonality, and one non-systematic component called noise. These components are defined as follows:

- Level: The average value in the series.
- Trend: The increasing or decreasing value in the series.
- Seasonality: The repeating short-term cycle in the series.
- Noise: The random variation in the series.

To generate the seasonal plot, an Air Passenger's data is imported. The data has the total number of passengers onboarded month-wise starting from 1949 to 1960. The topmost plot in Figure 4.20 depicts the variation in the number of passengers with respect to time. The subsequent plots show the decomposition of this original variation in terms of trend, seasonality, and noise.

```
1 from statsmodels.tsa.seasonal import seasonal_decompose
2
3 df=pd.read_csv('AirPassengers.csv')
4 seasonal_decompose(df).plot()
5
```



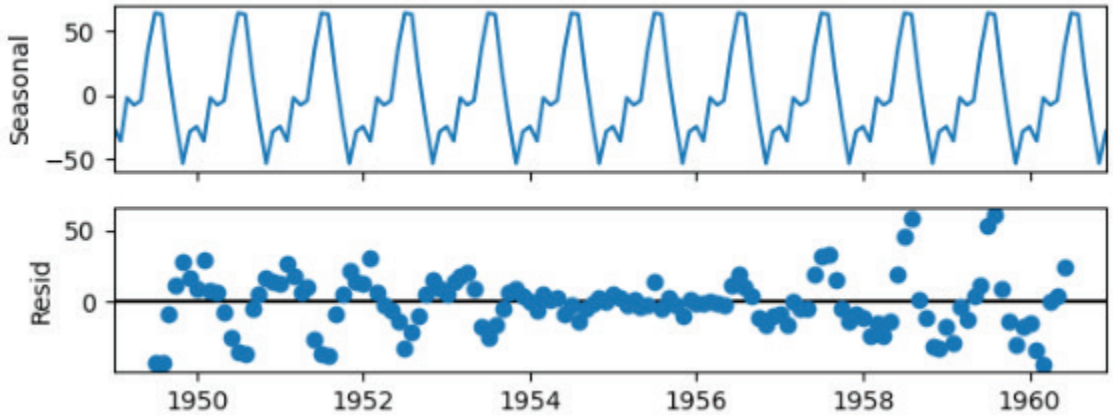


Fig. 4.20 Seasonal Decompose Plot for Air Passengers Data

Autocorrelation Function Plots (ACF Plots)

A statistical method known as the autocorrelation function (ACF) illustrates the correlation between values in a time series. The ACF plot demonstrates the relationship between a time series' present value and its historical values. The correlation coefficient is plotted against the lag, which is expressed as several periods or units, in the ACF plot, which is a bar chart. The correlation coefficient is expressed on the ACF plot's y-axis, while the number of lags is indicated on the x-axis. The number of delays required for a time series model can be determined using the ACF plot. It can also be used to determine the randomness and stationarity of a time series.

Partial Autocorrelation Function Plots (PACF Plots)

The partial correlation coefficients between a series and it's own lags are plotted in a PACF plot. The degree of correlation between two variables that cannot be accounted for by their mutual correlations with a predetermined group of other variables is known as the partial correlation. In contrast to an ACF plot, a PACF plot considers any correlation between observations with shorter lag times. The elements trend, seasonality, cyclicity, and residual can all be found in a time series. ACF considers each of these factors while determining correlations.

Both the original data and a model's residuals can be used for generating PACF graphs. They can aid in finding any significant moving average or autoregressive terms in the time series. The number of historical lags to incorporate in the forecasting equation of an auto-regressive model can also be determined using PACF charts. The model's Auto-Regression (AR) order is referred to as this.

```
from statsmodels.graphics.tsaplots import plot_acf,plot_pacf

plot_acf(df)
plt.title('ACF plot')
```

```
Text(0.5, 1.0, 'ACF plot')
```

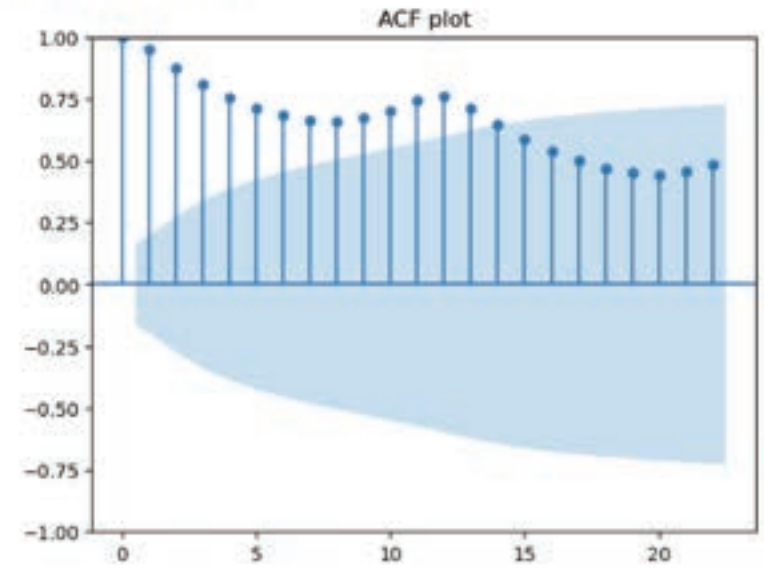


Fig. 4.21 A) ACF Plot for Air Passengers Data

```
from statsmodels.graphics.tsaplots import plot_acf,plot_pacf

plot_pacf(df)
plt.title('PACF plot')
```

```
Text(0.5, 1.0, 'PACF plot')
```

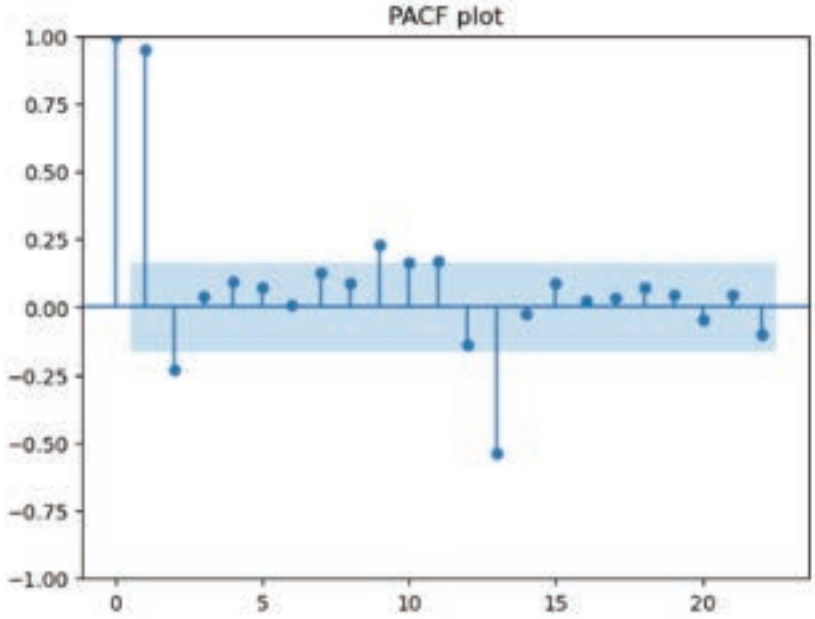


Fig. 4.21 B) ACF Plot for Air Passengers Data

4.2 CUSTOMIZATION AND ANNOTIZATION IN PLOT

Customization is kind of finishing the plot and making it representable and aesthetically good-looking so that viewers will understand it quickly and in a better way. Plots are created by Matplotlib using a built-in module called pyplot. There is a function called plot() in the pyplot module that accepts three arguments: the data set to be visualized, the type of visualization to create, and a collection of parameters to alter the plot. With the help of these adjustments, the user can alter the axis labels, reveal, or conceal the gridlines, establish the legends, and alter the colors of the data that is displayed on the graph. An example of giving the title and axis name is shown in the above figures.

```
import matplotlib.pyplot as plt

x = [1,2,3,4,5,6,7,8,9]
y = [1,4,9,16,25,36,49,64,81]

plt.figure(figsize=(4,2.25))
plt.plot(x,y,label='Quadratic Curve',
         color='blue', marker='o', linestyle='dashed',
     linewidth=2, markersize=5)
plt.xlabel('X')
plt.ylabel('Y')
plt.title('X vs Y')
plt.legend(loc='best')
plt.xticks(rotation=45)
plt.yticks(rotation=45)
plt.xlim(-2,12)
plt.ylim(-10,100)
```

```
(-10.0, 100.0)
```

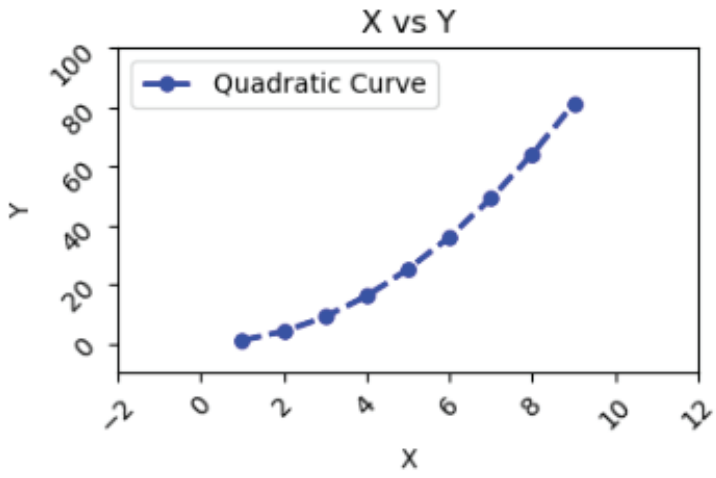


Fig. 4.22 Plot Customization

Annotation is like adding some extra spice to the plot title. The main purpose of the annotation is to write some information about the plot like its brief explanation or anything important that the user wants to mention.

```
import matplotlib.pyplot as plt

plt.figure(figsize=(5,4.5))
plt.plot([1,2,3], [1,4,9], color='blue', linestyle='dashed', marker='o')

a = 'This is an annotation used for briefing the plot'
plt.annotate(a, xy=(2.4, 5), xytext=(3.5, 5), arrowprops=dict(facecolor='blue', shrink=0.05))

```

```
Text(3.5, 5, 'This is an annotation used for briefing the plot')
```

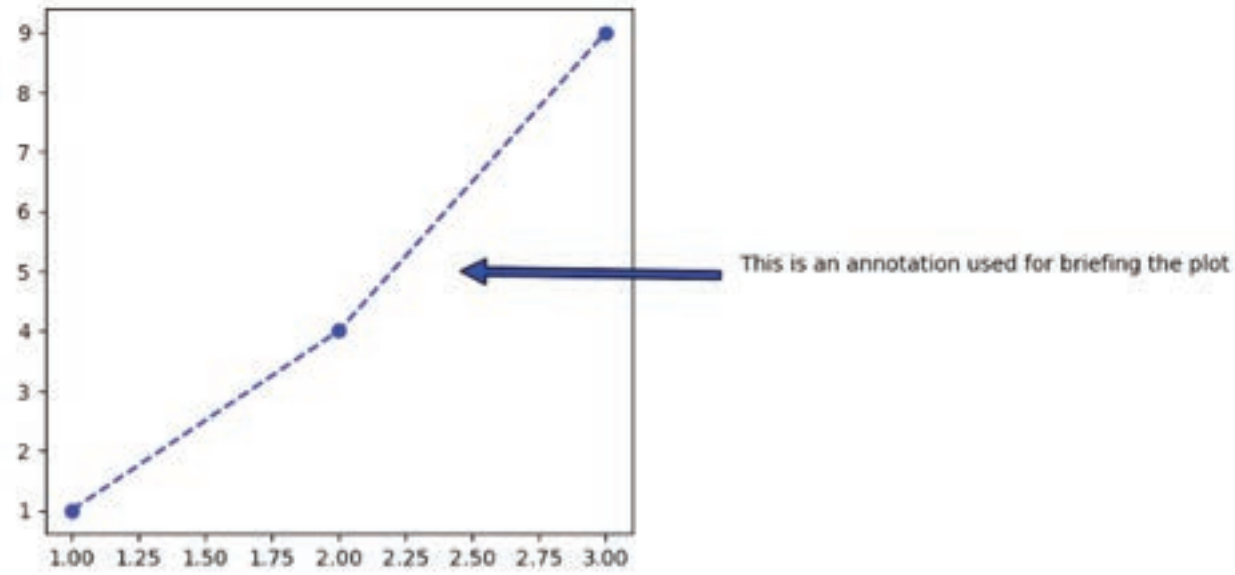


Fig. 4.23 Annotation of Plot

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 Which type of plot is best suited for visualizing the relationship between two continuous variables?
   a) Line plot
   b) Scatter plot
   c) Histogram
   d) Bar chart

2 What is the primary purpose of a scatter plot?
   a) Displaying the frequency distribution of data
   b) Showing the relationship between two or more variables
   c) Comparing categories or groups
   d) Visualizing the central tendency of data

3 In a bar chart, what type of variable is typically displayed on the x-axis?
a) Continuous variable
c) Categorical variable
b) Ordinal variable
d) Time variable

4 What type of time series plot is used to identify potential outliers or anomalies in the data?
a) Line plot
b) Scatter plot
c) Box plot
d) Autocorrelation plot

5 What does the length of the whiskers in a box plot represent?
a) The interquartile range (IQR) of the data
b) The range of the data
c) The mean of the data
d) The mode of the data

Answers

1: (b) *2: (b)* *3: (c)* *4: (c)* *5: (a)*

DESCRIPTIVE QUESTIONS

1 What are the key components of a scatter plot?
2 What are a line and histogram plot's fundamental uses?
3 Explain the key components of a Sunburst chart, including the central circle and the hierarchical rings.
4 What is the primary function of a Waterfall chart, and in what types of data analysis is it commonly used?
5 Describe the typical structure of a Pareto Chart, including the bars and the cumulative percentage line.

UNIT

3

KNIME

CHAPTER 1
ABOUT KNIME ANALYTICS PLATFORM

LEARNING OBJECTIVES

- Understand the primary purpose and significance of KNIME in data analytics and machine learning.
- Learn how to install KNIME on your system and configure it for your specific environment.
- Explore the essential steps involved in setting up KNIME, including plugin installation and directory selection.
- Familiarize yourself with the KNIME Workbench interface, its layout, and key functionalities.
- Discover how to navigate the Workbench and access essential tools and menus
- Identify and define critical components of KNIME, such as nodes, workflows, and data repositories.
- Understand the terminology unique to KNIME, ensuring clarity when working within the platform.

In today's digital age, data has become the lifeblood of decision-making and innovation. From business enterprises striving to understand customer behaviour to healthcare researchers deciphering complex genetic codes, the ability to harness and analyse data has never been more critical. This is where KNIME steps in as a game-changer.

KNIME Analytics Platform is a powerful and open-source data analytics, reporting, and integration platform. It is designed for data scientists, analysts, and engineers who need to perform data preprocessing, analysis, and visualization in a user-friendly and extensible environment. KNIME stands for "Konstanz Information Miner" and is developed by the KNIME AG, a software company headquartered in Zurich, Switzerland.

KNIME Logo

1.1 INTRODUCTION TO KNIME

KNIME Analytics Platform allows users to create data workflows, which are sequences of data processing steps. These workflows can be designed through a graphical user interface, making it accessible to users with varying levels of technical expertise. KNIME is particularly popular for its flexibility and ability to integrate with various data sources and tools, including databases, Python, R, and more.

Welcome to the world of KNIME, a versatile and open-source data analytics and integration platform that has transformed the way data professionals, scientists, and organizations approach data analysis. In this chapter, we'll embark on a journey to explore the essential concepts and features that make KNIME an indispensable tool in the realm of data science and analytics.

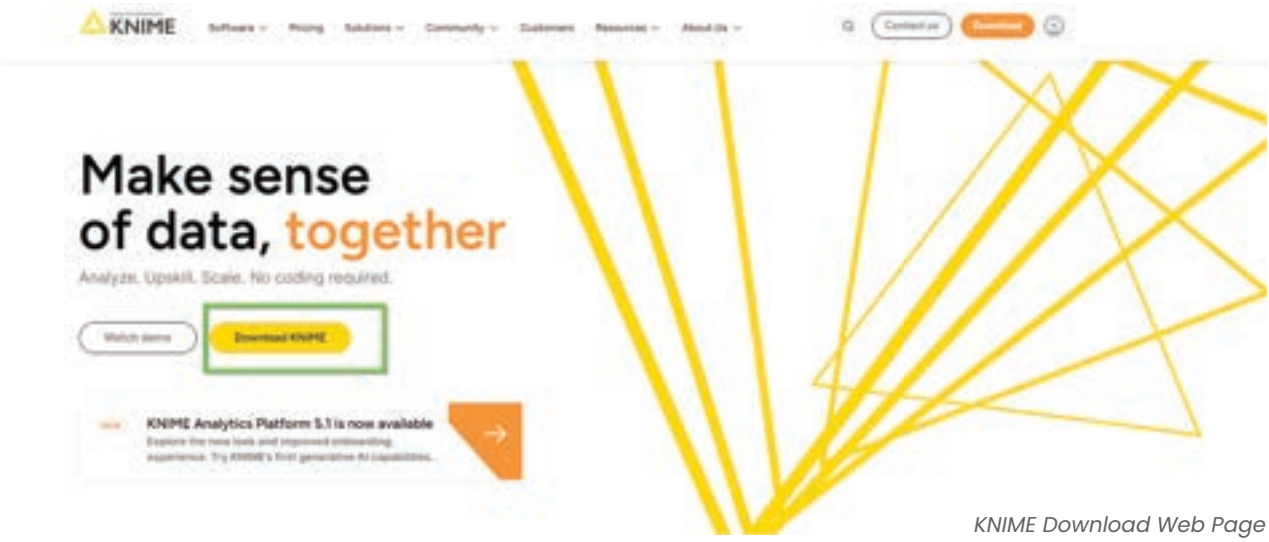
Key features and benefits of KNIME include:

1. **Open Source:** KNIME is open-source software, which means it is freely available and has a large community of users and contributors.
2. **Modular Workflow:** KNIME workflows are built using a modular approach. Users can drag and drop nodes to create workflows, making it easy to customize and expand data processing steps.
3. **Rich Node Repository:** KNIME provides a wide range of nodes for data manipulation, visualization, machine learning, and more. Users can extend the functionality by adding custom nodes or using community-contributed ones.
4. **Integration:** KNIME supports integration with various data formats, databases, programming languages, and other data analytics tools. This makes it a versatile choice for data analysis tasks.
5. **Scalability:** KNIME can handle both small and large datasets and is scalable to accommodate complex data analysis tasks.
6. **Automation:** It supports automation and batch processing, allowing users to schedule and run workflows at specified intervals.

1.2 KNIME INSTALLATION AND SETUP

To get started with KNIME Analytics Platform, you need to install it on your computer. Here's a general overview of the installation process:

Visit the KNIME Website:
Open your web browser and navigate to the official KNIME website at https://www.knime.com



KNIME Download Web Page

Choose Your Edition
KNIME offers two editions: KNIME Analytics Platform and KNIME Server. Choose the edition that suits your needs.

KNIME Analytics Platform: This is the free desktop version of KNIME, which is suitable for individual users and small teams. It allows you to create, execute, and manage data workflows.

KNIME Server: This is a commercial offering for larger organizations, offering features like collaboration, automation, and remote execution of workflows.

Download KNIME Analytics Platform:
If you want to download the free desktop version (KNIME Analytics Platform), click on the "Download" button associated with that edition. This will take you to the download page for the analytics platform.

Choose Your Version:
Select the version of KNIME Analytics Platform you want to download. Typically, you should choose the latest stable version.

Select Your Operating System:
Choose the appropriate version for your operating system (Windows, macOS, or Linux). Click on the download link for your OS.

Complete the Download:
Your browser will begin downloading the KNIME installer for your chosen operating system. The download may take a few minutes, depending on your internet speed.

Install KNIME:
Once the download is complete, locate the installer file (it's usually in your computer's Downloads folder) and double-click it to start the installation process. Follow the on-screen instructions to install KNIME on your computer.

Launch KNIME:
After the installation is complete, you can launch KNIME Analytics Platform. It may create a desktop shortcut, or you can find it in your Applications or Start menu.

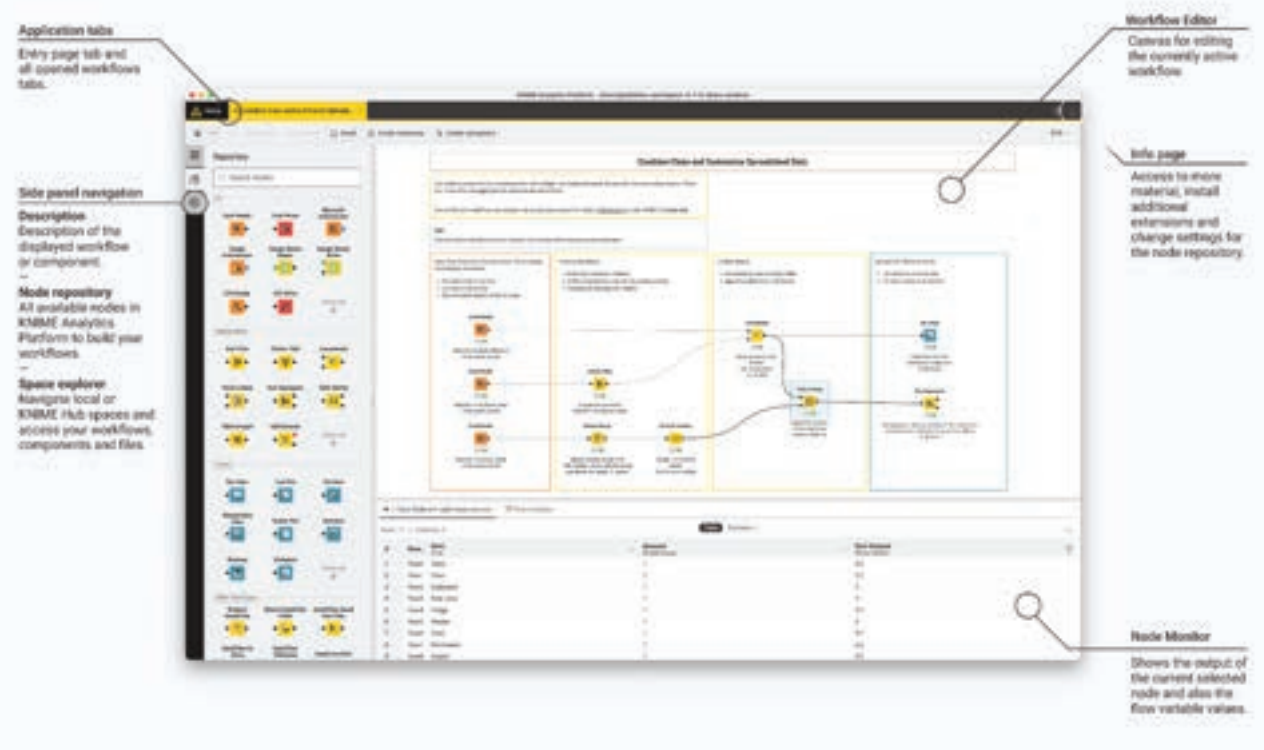
Start Using KNIME:
You can now start using KNIME to create data workflows, perform data analysis, and more. KNIME provides a user-friendly interface and a wide range of tools and extensions for data science and analytics tasks.

1.3 KNIME WORKBENCH OVERVIEW

Once KNIME is installed, you'll primarily interact with the KNIME Workbench, which is the main user interface for designing and executing data workflows. The Workbench consists of several key components:

1. **Workflow Editor:** This is the central area where you design your data workflows. You drag and drop nodes onto the canvas and connect them to define the flow of data processing. Also called Workspace.
2. **Node Repository:** The Node Repository is a panel where you can browse and search for available nodes. Nodes represent individual data processing or analysis operations.
3. **Console:** The Console provides feedback on the status of your workflows and any error messages. It's a useful tool for debugging.
4. **Outline:** The Outline panel gives you an overview of the structure of your workflow, showing the nodes and their connections.
5. **Node Description:** If a node is selected in the "Workflow Editor" or in the "Node repository," this panel displays a summary description of the selected node's functionalities.
6. **Explorer:** This panel shows the list of workflow projects available in the selected workspace(LOCAL) or on the EXAMPLES server or on other connected KNIME server.

   A sample figure is given as follows:

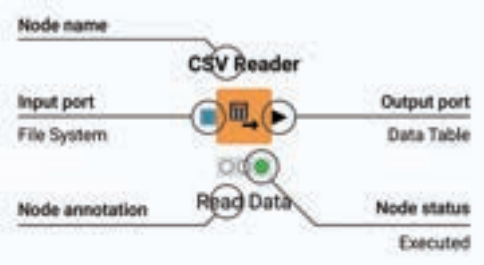


KNIME Workbench

1.4 KNIME COMPONENTS AND TERMINOLOGY

In KNIME, several key components and terminology are essential to understanding how workflows are constructed and executed:

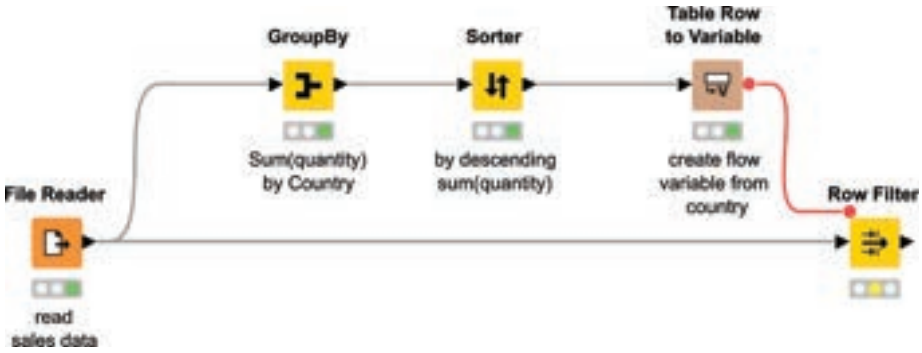
1 **Node:** Nodes are the building blocks of a KNIME workflow. Each node represents a specific operation or analysis step. For example, there are nodes for reading data from files, performing data transformations, running machine learning algorithms, and visualizing results. A sample figure is given as follows:



CSV Reader Node

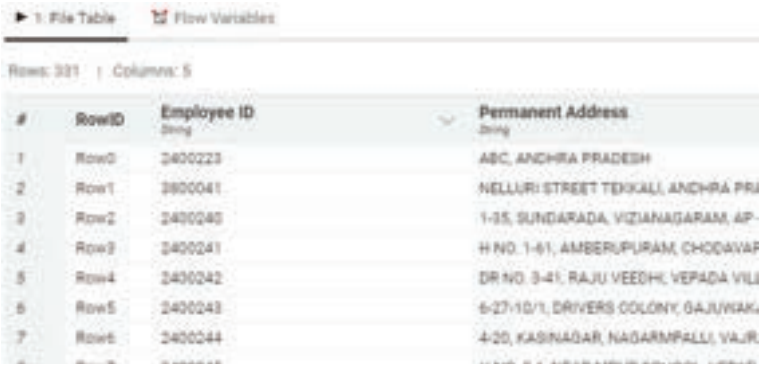
2 **Port:** Nodes have input and output ports. Data flows between nodes through these ports. Input ports receive data, and output ports send data to other nodes.

3 **Workflow:** A workflow is a sequence of nodes connected to each other. It represents the entire data analysis process, from data input to output. Workflows can be saved, reused, and shared. A sample figure is given as follows:



KNIME Workflow

4 **Data Table:** Data in KNIME is typically represented as a tabular data structure, similar to a spreadsheet. Each row is a data point, and each column is a feature or attribute. A sample figure is given as follows:



Data Table

5 **Connector:** Connectors are lines that link the output port of one node to the input port of another, defining the flow of data within a workflow. A sample figure is given as follows:

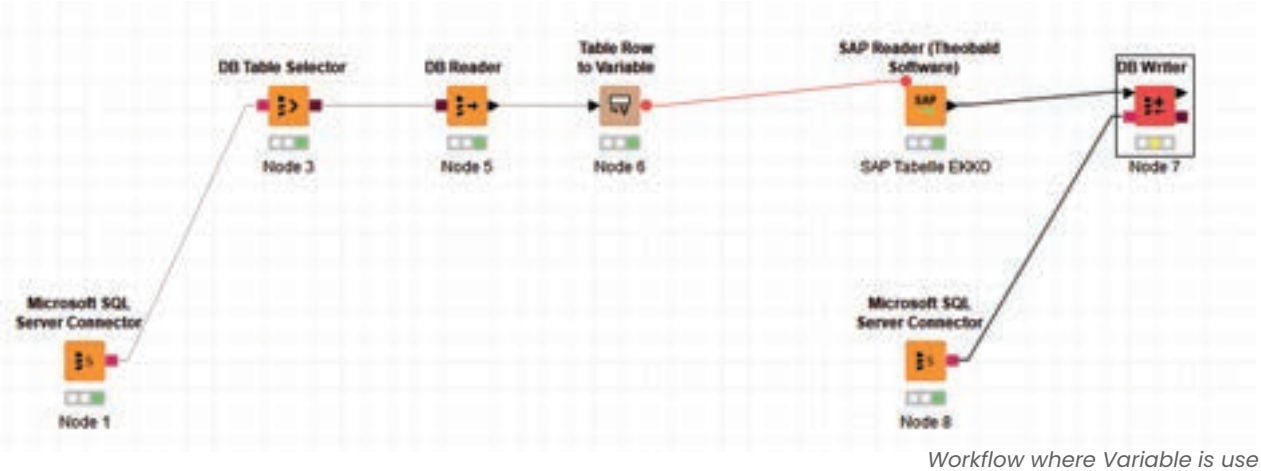
Different Connector

6 **Variable:** Variables are used to store and manage data or values within a workflow. They can be created, modified, and used in various nodes.

7 **Metanode:** A metanode is a container that allows you to group nodes and create reusable sub-workflows. It simplifies the visualization of complex workflows. A sample figure is given as follows:

Metanode

8 **Workflow Variable:** These are variables specific to a workflow and can be used to pass data or values between nodes within the same workflow. A sample figure is given as follows:



Workflow where Variable is use

SUMMARY

In conclusion, KNIME Analytics Platform serves as a powerful and versatile tool for data analytics and machine learning, providing users with a user-friendly Workbench interface to create, edit, and execute data workflows. Understanding the essential components and terminology of KNIME is vital for harnessing its full potential. The installation and setup process, as well as the utilization of nodes, workflows, and the Canvas, are fundamental aspects to grasp. By mastering these foundational elements, data scientists and analysts can efficiently explore, preprocess, and analyse data, making informed decisions and generating valuable insights to drive innovation and solve complex real-world problems. KNIME's flexibility and accessibility make it a valuable asset in the realm of data science and analytics.

REFERENCE

1 *https://www.knime.com/*
2 *https://www.knime.com/getting-started-guide*
3 *https://www.knime.com/knimepress*
4 *KNIME Beginner's Luck Book by Rosario_Silipo*
5 *KNIME Advanced Luck A Guide to KNIME Analytics Platform for Advanced Users by Rosario Silipo, Jeanette Prinz*

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 What is KNIME Analytics Platform primarily designed for?
   a) Video editing
   b) Data analytics and machine learning
   c) Music production
   d) Web development

2 Which of the following is a step typically involved in KNIME installation and setup?
   a) Choosing a data visualization tool
   b) Installing a web browser
   c) Selecting an installation directory
   d) Setting up an email account

3 In KNIME, what does the "Workbench" refer to?
   a) A physical bench where you work
   b) The main interface for creating and editing workflows
   c) A tool for web development
   d) A software development environment

4 What are nodes in KNIME?
   a) Nodes are celestial bodies in space.
   b) Nodes are individual processing units that perform specific tasks in a workflow.
   c) Nodes are elements of a musical composition.
   d) Nodes are web browser tabs.

5 What file format is commonly used for saving KNIME workflows?
   a) .doc
   b) .pdf
   c) .knar
   d) .jpg

6 What is a typical use case for KNIME workflows?
   a) Sending emails
   b) Creating 3D animations
   c) Data preprocessing and analysis
   d) Writing code in multiple programming languages

7 In KNIME, what is a component that stores information about data sources, files, and settings?
   a) Node
   b) Workflow
   c) Data Repository
   d) Canvas

8 What does the term "Terminology" refer to in the context of KNIME?
   a) A glossary of terms used within the platform
   b) The termination of a workflow
   c) The software's shutdown process
   d) A programming language used in KNIME

9 Which menu in KNIME allows you to access various data preprocessing and analysis tools?
   a) File
   b) Edit
   c) Data
   d) Help

10 What is the primary function of the "Workflow Editor" in KNIME?
   a) Editing images
   b) Writing poetry
   c) Creating and modifying workflows
   d) Editing videos

11 What is the purpose of the "Node Repository" in KNIME?
   a) To store physical objects
   b) To organize and access nodes for building workflows
   c) To create new workflows
   d) To browse the internet

12 What does the "Canvas" represent in KNIME?
   a) A type of painting material
   b) The main area where you design workflows
   c) A type of fabric
   d) A web browser interface

13 Which of the following is NOT a typical component of KNIME?
   a) Node
   b) Workflow
   c) Spreadsheet
   d) Database

14 What is the recommended first step when setting up KNIME Analytics Platform?
   a) Open a new blank project
   b) Install additional plugins
   c) Choose the installation directory
   d) Create a new email account

15 Which of the following is NOT a common use of KNIME Analytics Platform?
   a) Predictive analytics
   b) Data visualization
   c) Video editing
   d) Data cleansing

Answers

1 b) *Data analytics and machine learning*
2 c) *Selecting an installation directory*
3 b) *The main interface for creating and editing workflows*
4 b) *Nodes are individual processing units that perform specific tasks in a workflow.*
5 c) *.knar*
6 c) *Data preprocessing and analysis*
7 c) *Data Repository*
8 a) *A glossary of terms used within the platform*
9 c) *Data*
10 c) *Creating and modifying workflows*
11 b) *To organize and access nodes for building workflows*
12 b) *The main area where you design workflows*
13 d) *Database*
14 c) *Choose the installation directory*
15 c) *Video editing*

CHAPTER 2
VISUAL KNIME WORKFLOWS

LEARNING OBJECTIVES

- Understand the essential steps in constructing KNIME workflows visually.
- Learn to add, configure, and connect nodes to create data processing workflows.
- Develop proficiency in organizing nodes to create logical data flow.
- Familiarize yourself with the KNIME Workflow Editor interface.
- Learn to navigate and use editor features for efficient workflow design.
- Understand how data is passed between nodes and learn to inspect intermediate and final results.
- Learn to configure workflow settings, including data sources and node parameters.
- Explore workflow execution options and result generation.

KNIME Analytics Platform is known for its visual approach to creating data workflows. Visual workflows allow users to design, manipulate, and execute data processing tasks without writing code. Here's an overview of working with visual KNIME workflows:

2.1 BUILDING A BASIC WORKFLOW

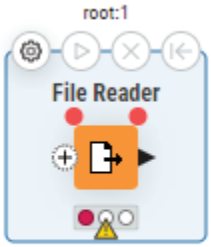
1. **Launching KNIME:** After installing KNIME, launch the application.
2. **Creating a New Workflow:** To start building a workflow, go to the "File" menu and select "New Workflow." This creates a blank canvas where you can design your workflow. A sample figure is given as follows:

Create New Workflow

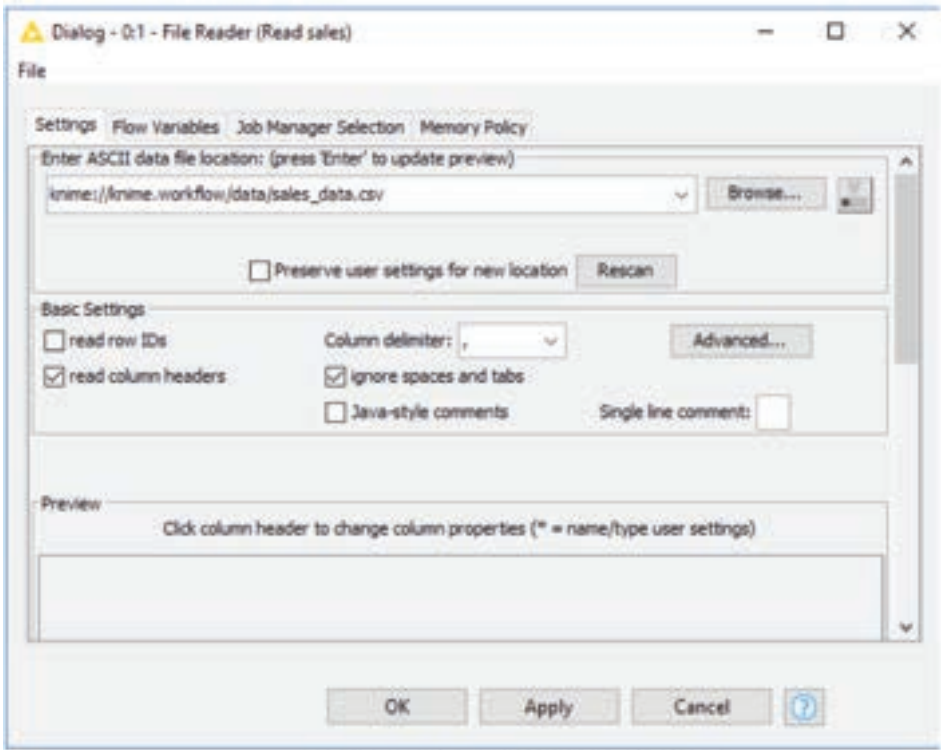
3 **Adding Nodes:** Workflows in KNIME are built by adding nodes. Nodes represent specific data processing tasks. You can add nodes from the Node Repository by dragging and dropping them onto the canvas.

4 **Connecting Nodes:** Nodes are connected using connectors. To establish a data flow, connect the output port of one node to the input port of another. This defines the order in which data is processed.

5 **Configuring Nodes:** Double-click on a node to open its configuration dialog. Here, you can specify parameters and settings for the node. The configuration options vary depending on the type of node. A sample figure is given as follows:



File Reader Node



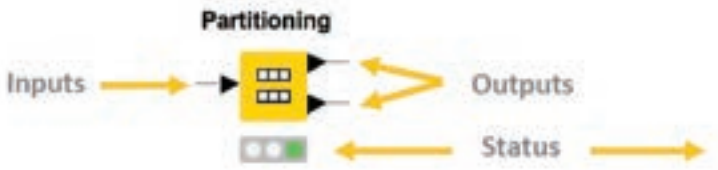
File Reader Node Configuration screen

6 **Running the Workflow:** To execute the workflow, click the "Run" button on the toolbar. KNIME will start processing the data according to the defined workflow.

2.1 DATA FLOW IN KNIME

Data flow is a fundamental concept in KNIME. It defines how data is processed and transformed as it moves through the workflow. Key points about data flow in KNIME include:

1 **Input and Output Ports:** Each node in KNIME has input and output ports. Data flows into a node through its input port, and the results flow out through its output port. A sample figure is given as follows:

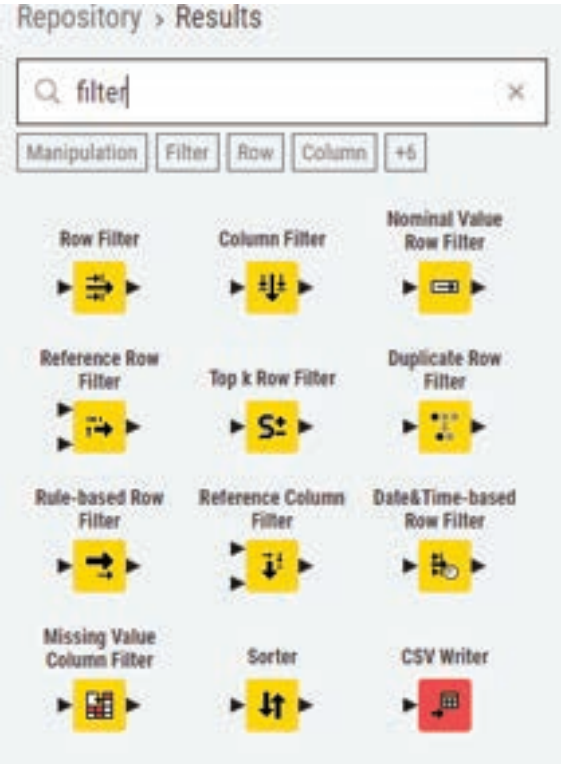


Input Output Ports to node with executed status

2 **Connectors:** Connectors are used to link the output port of one node to the input port of another. This connection defines the order in which nodes are executed.

3 **Data Table:** Data in KNIME is typically represented as a Data Table. Each row in the table corresponds to a data point, and each column represents a feature or attribute.

4 **Data Transformation:** Nodes in the workflow perform various data transformations, such as filtering, aggregation, or machine learning. Data is transformed as it passes through these nodes. A sample figure is given as follows:



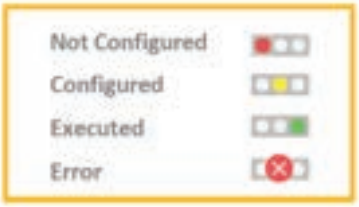
Filter related Node examples

Transform related Node examples

2.2 WORKFLOW CONFIGURATION AND EXECUTION

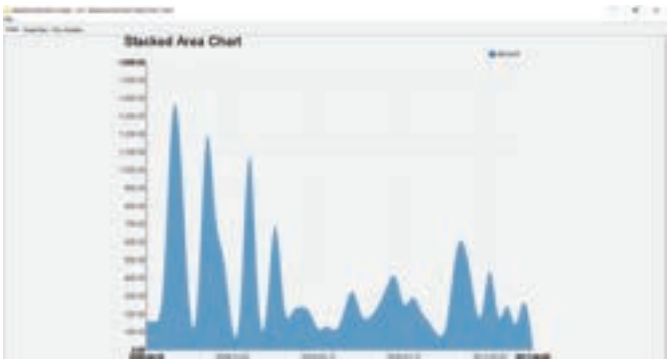
Configuring and executing workflows in KNIME involves the following steps:

1. **Node Configuration:** Double-click on a node to open its configuration dialog. Here, you can set parameters, choose data sources, and specify other settings relevant to the node's operation.
2. **Variable Assignment:** You can use variables to pass data or values between nodes within the same workflow. Assign variables in nodes' configuration dialogs.
3. **Execution Control:** Click the "Run" button to execute the workflow. KNIME will process the data according to the defined workflow logic.
4. **Monitoring Execution:** Keep an eye on the Console for progress updates, and if any issues or errors arise during execution, KNIME will provide feedback there. A sample figure is given as follows:

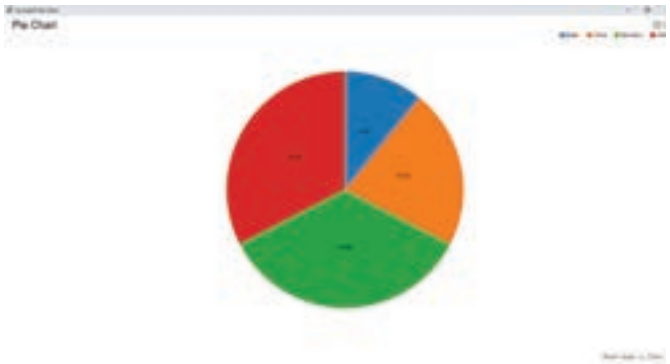


Different Execution Status

5. **Workflow Results:** After execution, the workflow may produce results, such as visualizations or data tables. These can be viewed and saved as needed. A sample figure is given as follows:



Stacked Area Chart Sample



Pie-Chart Sample

KNIME's visual workflow approach, coupled with its powerful data processing capabilities, makes it a versatile tool for data analysts and scientists, allowing them to create complex data workflows with ease.

EXERCISE

Create empty workflow

- Click "New" in the toolbar panel
- Right Clicking a folder in the local workspace in the KNIME Explorer
- Enter the name of the "WorkFlow"
- Browse the destination folder
- Click "Finish"

SUMMARY

In conclusion, mastering the creation and management of visual KNIME workflows is an essential skill for data analysts and scientists. These workflows serve as powerful tools for organizing, processing, and deriving insights from data in a visually intuitive manner. Understanding how to build workflows, navigate the Workflow Editor interface, manage data flow, and configure and execute workflows empowers individuals to efficiently perform data analysis tasks within the KNIME platform. By achieving proficiency in these areas, users can harness the full potential of KNIME to streamline their data-driven processes, make informed decisions, and generate valuable insights for a wide range of applications, from data preprocessing to advanced analytics and machine learning. Visual KNIME workflows not only enhance productivity but also contribute to more effective data-driven decision-making in various domains, making them a valuable asset in the toolkit of data professionals.

REFERENCE

1 *https://www.knime.com/*
2 *https://www.knime.com/getting-started-guide*
3 *https://www.knime.com/knimepress*
4 *KNIME Beginner's Luck Book by Rosario_Silipo*
5 *KNIME Advanced Luck A Guide to KNIME Analytics Platform for Advanced Users by Rosario Silipo, Jeanette Prinz*

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 What is the first step in building a basic KNIME workflow?
   a) Adding nodes
   b) Configuring nodes
   c) Connecting nodes
   d) Running the workflow

2 In KNIME, what is the purpose of connecting nodes in a workflow?
   a) To rearrange nodes
   b) To create visual designs
   c) To establish data flow
   d) To change node colors

3 What is the benefit of organizing nodes logically in a KNIME workflow?
   a) It improves the aesthetic appeal.
   b) It enhances node performance.
   c) It creates clear data flow.
   d) It adds animation to the workflow.

4 Where do you typically design and edit KNIME workflows?
   a) Spreadsheet
   b) Workflow Editor
   c) Web browser
   d) Email client

5 What can you do to customize the appearance of a KNIME workflow in the editor?
   a) Change the font size
   b) Apply filters to the data
   c) Adjust the canvas size
   d) Send the workflow as an email attachment

6 Which part of the Workflow Editor interface allows you to search for nodes and components?
   a) The Canvas
   b) The Toolbar
   c) The Node Repository
   d) The Node Settings panel

7 In KNIME, what does data flow refer to?
   a) The movement of data between nodes in a workflow
   b) The speed of data processing
   c) The color of data visualizations
   d) The file format used to store data

8 What happens when you connect the output of one node to the input of another node in KNIME?
   a) A mathematical equation is formed.
   b) Data is transferred from one node to another.
   c) The workflow becomes invalid.
   d) The nodes change color.

9 Which of the following is a common data manipulation task in KNIME?
   a) Browsing the internet
   b) Playing audio files
   c) Filtering and transforming data
   d) Editing images

10 How can you configure the input data for a KNIME workflow?
   a) By changing the canvas color
   b) By configuring node parameters
   c) By sending an email
   d) By adding more nodes

11 What is the purpose of executing a KNIME workflow?
   a) To change the workflow's layout
   b) To create a new workflow
   c) To generate results based on the workflow's logic
   d) To edit node settings

12 How can you save and export a KNIME workflow for sharing with others?
   a) By taking a screenshot
   b) By printing it on paper
   c) By using the "File" menu options
   d) By copying and pasting nodes

Answers

1 *a) Adding nodes*
2 *c) To establish data flow*
3 *c) It creates clear data flow.*
4 *b) Workflow Editor*
5 *c) Adjust the canvas size*
6 *c) The Node Repository*
7 *a) The movement of data between nodes in a workflow*
8 *b) Data is transferred from one node to another.*
9 *c) Filtering and transforming data*
10 *b) By configuring node parameters*
11 *c) To generate results based on the workflow's logic*
12 *c) By using the "File" menu options*

CHAPTER 3
DATA ACCESS

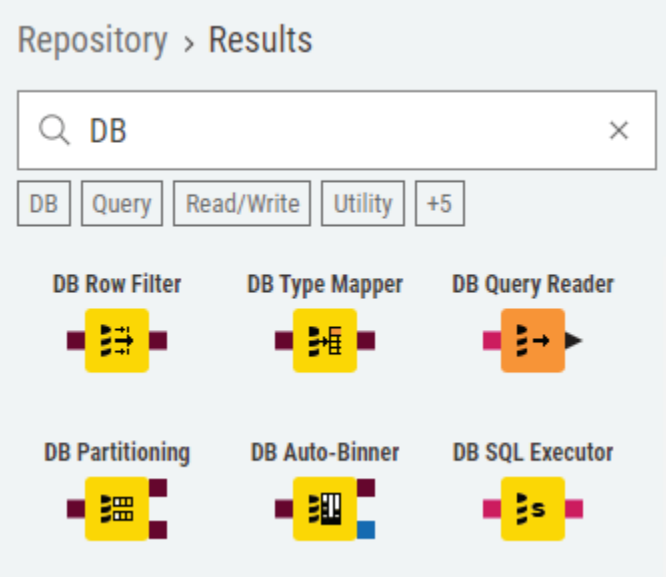
LEARNING OBJECTIVES

- Efficiently import data from various sources into KNIME, configuring import settings and handling data formats.
- Perform data preprocessing and cleaning within KNIME, addressing issues like missing data, outliers, and duplicates.
- Develop proficiency in data integration, combining multiple sources, and transforming data for analysis.
- Understand methods and options for exporting cleaned and transformed data from KNIME for further use.
- Master techniques to enhance data quality and usability while working with diverse datasets.
- Enable effective data sharing and reporting by saving data in various formats and across different tools and platforms.

Data access is a crucial step in any data analysis process. In KNIME, you can access data from various sources, prepare it for analysis, and export the results. Here's how it works:

3.1 IMPORTING DATA INTO KNIME

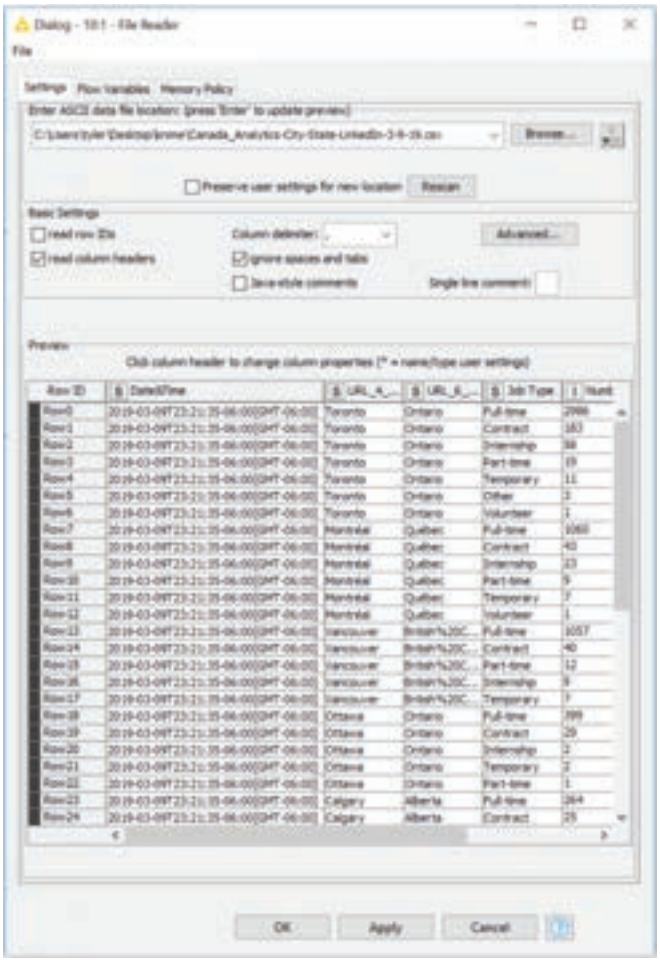
1 **Data Source Selection:** To import data into KNIME, open a new or existing workflow. Start by selecting a suitable data source. KNIME supports a wide range of data sources, including CSV files, Excel spreadsheets, databases (e.g., MySQL, PostgreSQL), web services, and more. A sample figure is given as follows:



Data Source related Node examples

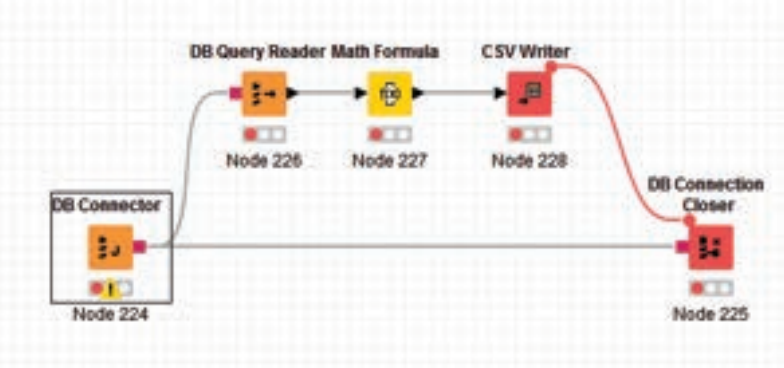
2 **File Reader Node:** If you're working with a file-based data source like CSV or Excel, you can use the "File Reader" node. Configure the node to specify the file location and format options. After configuring, the node reads the data and creates a data table. A sample figure is given as follows:

File Reader Node

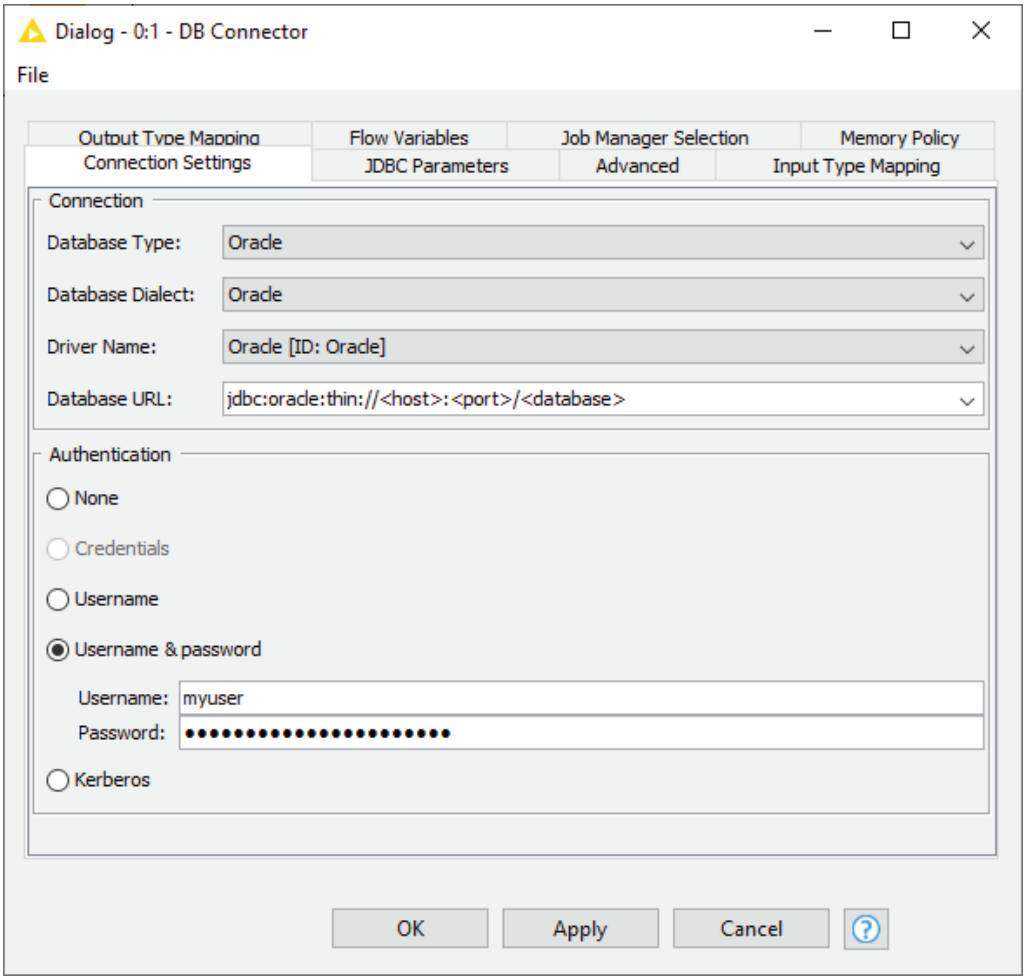


File Reader Node Configuration screen

3 **Database Connector Node:** When working with databases, use the "Database Connector" node to establish a connection to the database. You'll need to provide connection details such as the database URL, username, and password. Once connected, you can write SQL queries to retrieve data. A sample figure is given as follows:

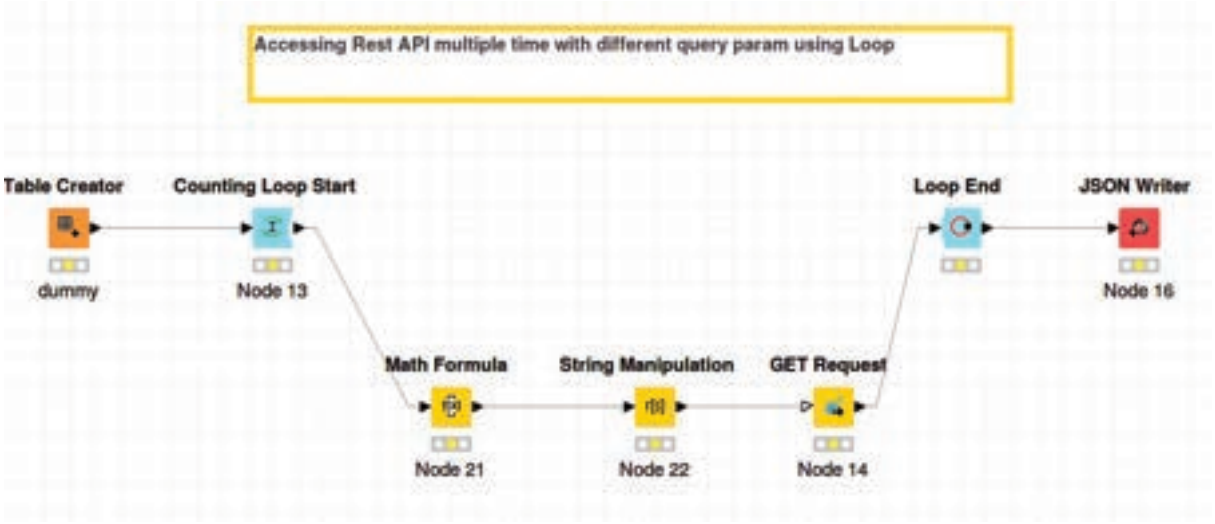


Database Connector Node with workflow



Database Connector Node configuration screen

4 **Other Data Import Nodes:** KNIME offers additional nodes for specific data sources, such as REST API nodes for web data, image processing nodes for image data, and more. Choose the appropriate node based on your data source. A sample figure is given as follows:



REST API Node with workflow

3.2 DATA PREPARATION AND CLEANING

Data preparation and cleaning are essential steps to ensure that your data is accurate and ready for analysis. KNIME provides various nodes and tools for these tasks:

1 **Data Exploration:** Use nodes like "Data Explorer" to get an initial overview of your data. This node generates summary statistics and visualizations. A sample figure is given as follows:

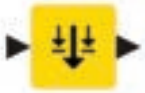
Data Explorer Node

2 **Data Cleaning:** Nodes like "Missing Value" and "Duplicate Row Filter" help you handle missing data and remove duplicate records. A sample figure is given as follows:

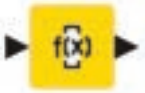
Missing Value Node

Duplicate Row Filter Node

3 **Data Transformation:** You can transform data using nodes like "Column Filter," "Math Formula," and "String Manipulation" to perform operations such as renaming columns, calculating new variables, and formatting data. A sample figure is given as follows:



Column Filter Node

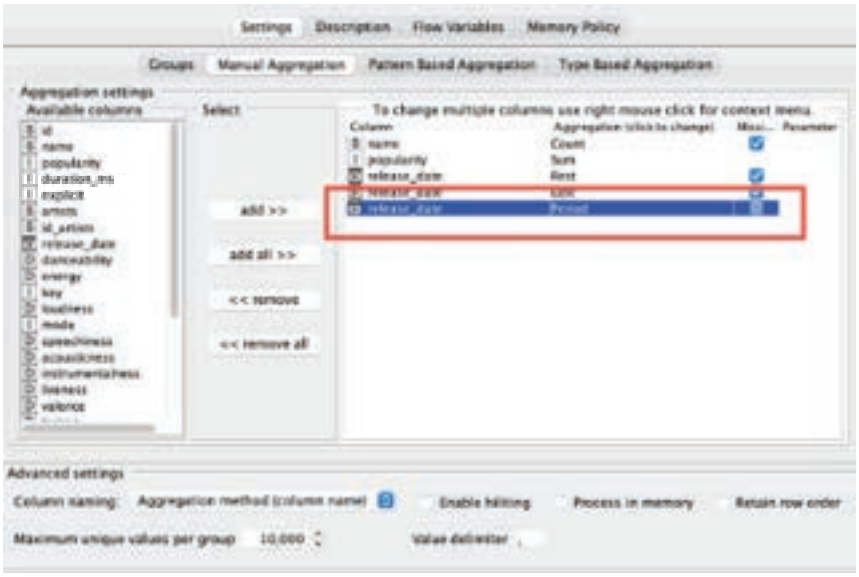


Math Node

String Manipulation Node

4 **Data Aggregation:** Use nodes like "GroupBy" to perform aggregation operations on your data, such as calculating sums, averages, or counts. A sample figure is given as follows:

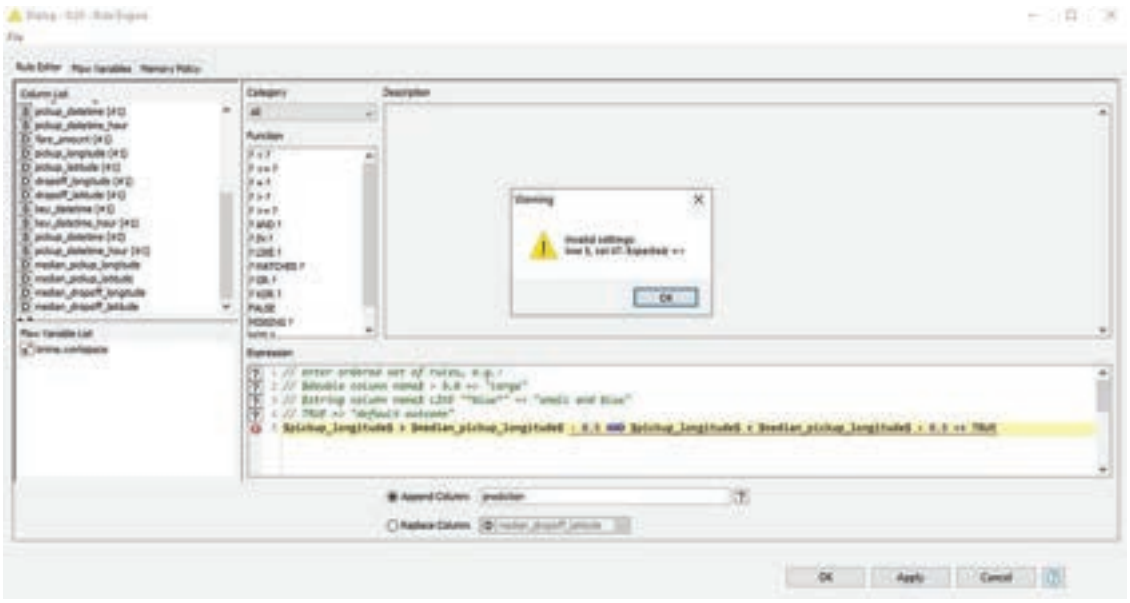
GroupBy Node



GroupBy Node Configuration

5 **Data Imputation:** KNIME provides nodes for imputing missing values, such as the "Missing Value" and "Rule Engine" nodes. A sample figure is given as follows:

Rule Engine Node

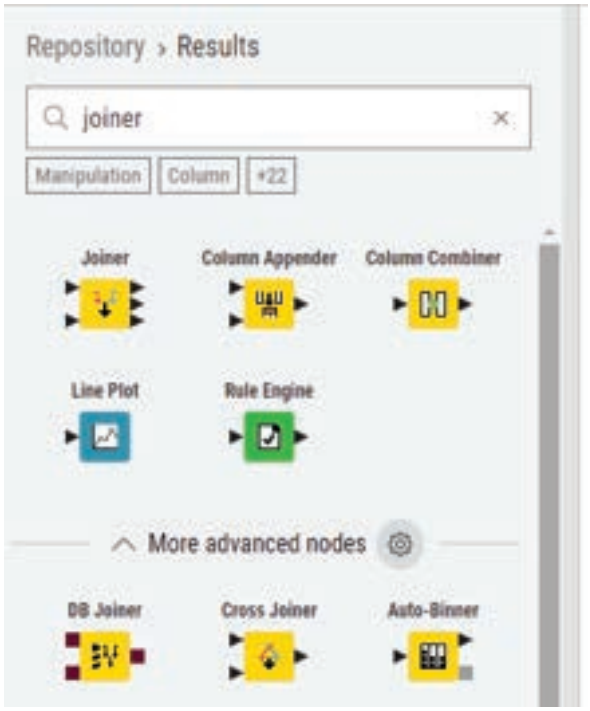


Rule Engine Node Configuration

3.3 DATA INTEGRATION AND TRANSFORMATION

Data integration and transformation involve combining data from different sources, reshaping it, and preparing it for analysis. KNIME offers a wide range of nodes for these purposes:

1 **Joining Data:** Use nodes like "Joiner" to combine data from multiple tables based on common keys or criteria. A sample figure is given as follows:



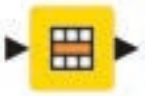
Joiner Node examples

2 **Pivoting and Unpivoting:** Nodes like "Pivoting" and "Unpivoting" help reshape data from wide to long format or vice versa. A sample figure is given as follows:

Unpivoting Node

Pivoting Node

3 **Data Sampling:** You can use nodes like "Row Sampling" to select a subset of your data for analysis. A sample figure is given as follows:



Row Sampling Node

4 **Data Normalization and Scaling:** Normalize and scale your data using nodes like "Normalizer" and "Scorer" to prepare it for machine learning algorithms. A sample figure is given as follows:

Normalizer Node

Scorer Node

5 **Text Mining and NLP:** KNIME also supports text data processing and natural language processing (NLP) with nodes like "Text Preprocessing" and "Bag of Words." A sample figure is given as follows:

Text Preprocessing Node

Bag of Words Node

3.4 EXPORTING DATA FROM KNIME

Once you have analysed and transformed your data in KNIME, you can export the results for further use or reporting:

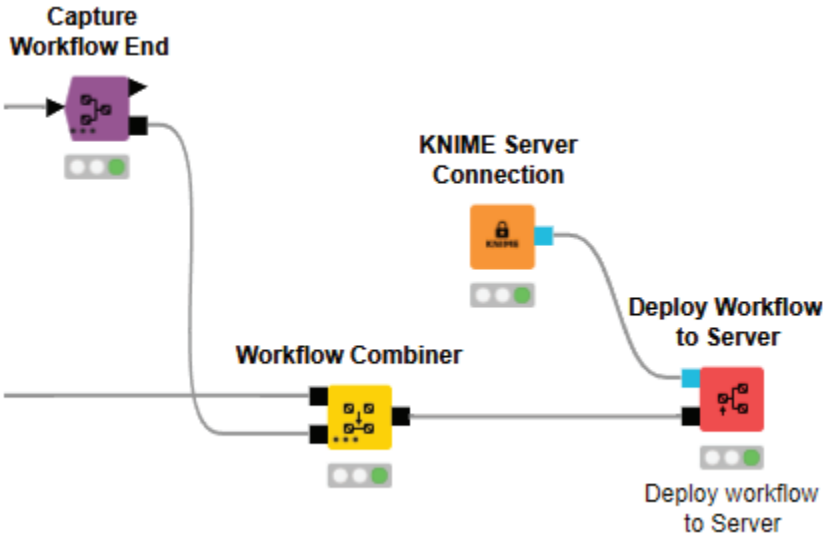
1 **Data Export Node:** To export data, you can use nodes like "CSV Writer," "Excel Writer," or "Database Writer," depending on your desired output format. A sample figure is given as follows:

Data Export related Node examples

2 **Data Visualization:** KNIME provides various nodes for data visualization, allowing you to create charts, plots, and visual summaries of your data. You can export these visualizations as images or reports. A sample figure is given as follows:

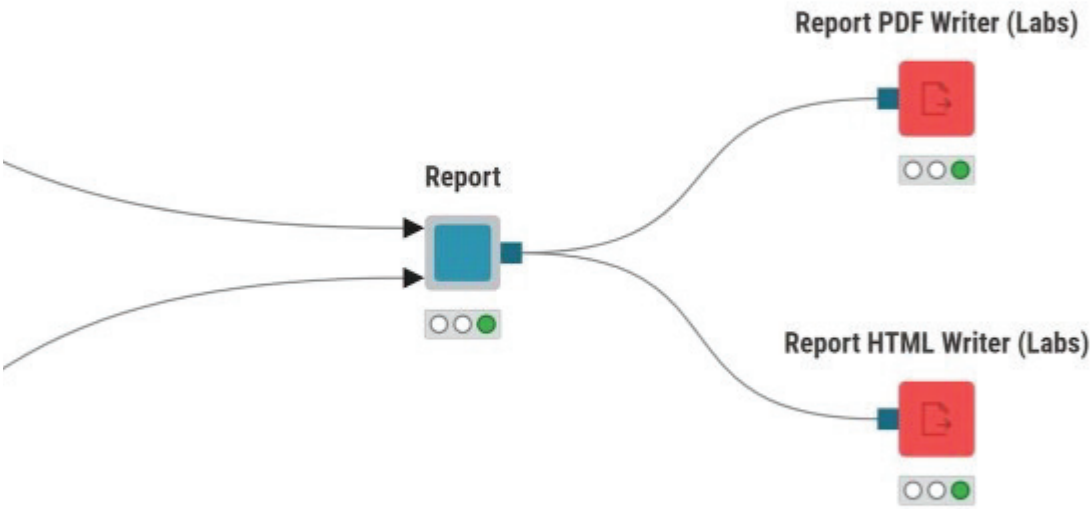
Data Export related Node examples

3 **Model Deployment:** If you've built machine learning models in KNIME, you can export them for deployment in production environments. A sample figure is given as follows:



Deploy Workflow to server example

4 **Data Reports:** KNIME also allows you to create reports with customized layouts and export them in various formats, such as PDF or HTML. A sample figure is given as follows:



Data Report related Node example

KNIME's flexibility in data access, preparation, integration, and export makes it a powerful tool for managing the entire data analysis pipeline, from data acquisition to insights generation.

EXERCISE

Importing Data into KNIME:

- Launch KNIME Analytics Platform on your computer.
- Create a new workflow in the Workflow Editor.
- Use the Node Repository to add a "File Reader" node to your workflow.
- Configure the "File Reader" node to import a sample dataset (e.g., a CSV file) from your local storage.
- Execute the workflow to load the data into KNIME.

Data Preparation and Cleaning:

- Add a "Missing Value" node to your workflow.
- Connect the "File Reader" node to the "Missing Value" node.
- Configure the "Missing Value" node to handle missing values in the dataset (e.g., by imputing them with mean values).
- Add a "Data Row Filter" node to filter out specific rows based on a condition (e.g., filter out rows where a specific column value is below a threshold).
- Execute the workflow to apply data preparation and cleaning steps.

Data Integration and Transformation:

- Add a "Joiner" node to your workflow.
- Connect the "Missing Value" node and the "Data Row Filter" node to the "Joiner" node.
- onfigure the "Joiner" node to perform a join operation between two data streams (e.g., joining datasets by a common column).
- Add a "Column Filter" node to select specific columns from the joined dataset.
- Execute the workflow to integrate, transform, and select columns in the data.

Exporting Data from KNIME:

- Add a "CSV Writer" node to your workflow.
- Connect the "Column Filter" node to the "CSV Writer" node.
- Configure the "CSV Writer" node to specify the output file location and format.
- Execute the workflow to export the transformed data to a new CSV file.

Validation and Reflection:

- Open the exported CSV file to verify that the data has been successfully cleaned, integrated, transformed, and exported.
- Reflect on your experience with data access and manipulation in KNIME.
- Note any challenges you encountered and how you resolved them.
- Consider how these data access and transformation techniques can be applied to real-world data analysis tasks.

This exercise will provide you with hands-on experience in using KNIME to import, prepare, clean, integrate, transform, and export data, which are essential skills for data professionals and analysts.

SUMMARY

In conclusion, mastering the art of data access is foundational to the success of any data-driven endeavour. The ability to efficiently import, prepare, and clean data within KNIME not only ensures the integrity and quality of the data but also sets the stage for meaningful analysis and insights. Moreover, data integration and transformation techniques empower data professionals to harmonize disparate data sources, making them a valuable asset in the decision-making process. Equally important is the capability to export data in various formats, enabling seamless collaboration and reporting with stakeholders and other data analysis tools. By achieving these objectives, individuals are better equipped to navigate the complexities of data access, ensuring that data serves as a reliable foundation upon which to build valuable insights and drive informed decisions across a wide spectrum of industries and applications.

REFERENCE

1 *https://www.knime.com/*
2 *https://www.knime.com/getting-started-guide*
3 *https://www.knime.com/knimepress*
4 *KNIME Beginner's Luck Book by Rosario_Silipo*
5 *KNIME Advanced Luck A Guide to KNIME Analytics Platform for Advanced Users by Rosario Silipo, Jeanette Prinz*

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 How can you import data into KNIME?
   a) By using a web browser
   b) By sending an email
   c) By configuring node parameters
   d) By changing the canvas size

2 What is the purpose of configuring data import settings in KNIME?
   a) To change the color of nodes
   b) To handle missing data
   c) To create visual designs
   d) To rearrange nodes

3 Which of the following is a common source for importing data into KNIME?
   a) A video game console
   b) A music player
   c) An external spreadsheet files
   d) A digital camera

4 What is the objective of data cleaning in KNIME?
   a) To introduce errors into the data
   b) To remove duplicate data points
   c) To generate more data
   d) To change the canvas color

5 How can you address missing data in KNIME?
   a) By ignoring it and proceeding with the analysis
   b) By replacing missing values with appropriate data
   c) By deleting the entire dataset
   d) By changing the font size

6 What is a common task in data preparation in KNIME?
   a) Playing audio files
   b) Handling outliers
   c) Creating 3D animations
   d) Sending emails

7 In KNIME, what does data integration involve?
   a) Splitting data into smaller parts
   b) Combining data from multiple sources
   c) Deleting data
   d) Changing node colors

8 What is data transformation in KNIME?
   a) The process of converting data into images
   b) The process of modifying data for analysis
   c) The process of sending data to external devices
   d) The process of creating websites

9 Which of the following is a common data transformation task in KNIME?
   a) Playing video games
   b) Painting pictures
   c) Aggregating data
   d) Browsing the internet

10 What is the purpose of exporting data from KNIME?
   a) To change the canvas layout
   b) To create new data
   c) To save data for use in other tools or reports
   d) To edit node settings

11 How can you share data with others effectively from KNIME?
   a) By taking a screenshot of the data
   b) By printing it on paper
   c) By exporting it in various formats
   d) By playing a video

12 Which menu option in KNIME allows you to export data?
   a) File
   b) Edit
   c) Data
   d) Help

Answers

1 c) *By configuring node parameters*
2 b) *To handle missing data*
3 c) *An external spreadsheet files*
4 b) *To remove duplicate data points*
5 b) *By replacing missing values with appropriate data*
6 b) *Handling outliers*
7 b) *Combining data from multiple sources*
8 b) *The process of modifying data for analysis*
9 c) *Aggregating data*
10 c) *To save data for use in other tools or reports*
11 c) *By exporting it in various formats*
12 a) *File*

CHAPTER 4
BIG DATA

LEARNING OBJECTIVES

- Learn to establish connections to various Big Data sources within KNIME, enabling data access and retrieval from distributed systems.
- Acquire the skills to efficiently handle and process large-scale data within the KNIME platform, optimizing performance and resource utilization.
- Understand the principles of distributed data processing, including parallelization and scalability, to harness the potential of Big Data in analytical workflows.
- Develop proficiency in using KNIME for Big Data analytics, including data exploration, modeling, and generating actionable insights from massive datasets.

Leveraging KNIME for Big Data analytics can be a powerful way to analyse large and complex datasets efficiently.

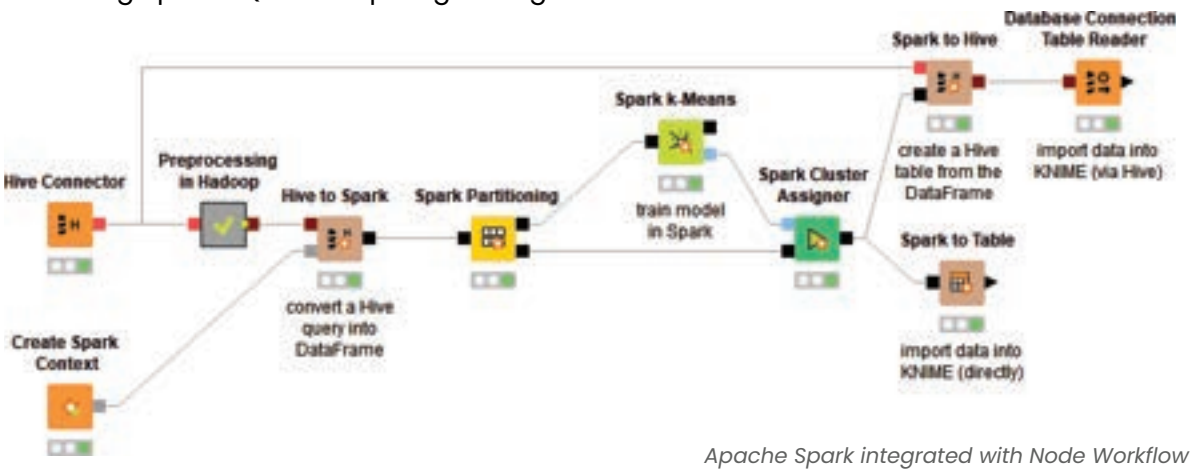
4.1 CONNECTING TO BIG DATA SOURCES

Connecting to Big Data sources is a crucial step in leveraging KNIME for Big Data analytics. KNIME provides various connectors and integrations to connect to Big Data platforms:

1 **Hadoop Distributed File System (HDFS):** KNIME supports connecting to HDFS, a common storage system in the Hadoop ecosystem. You can use the "HDFS Connection" and "HDFS File Picker" nodes to read and write data from and to HDFS. A sample figure is given as follows:

HDFS File Picker Node

2 **Apache Spark:** KNIME integrates with Apache Spark, a powerful Big Data processing framework. You can use the "Spark Reader" node to load data from Spark Data Frames into KNIME workflows. Similarly, you can use the "Spark SQL" node for querying and manipulating data using Spark SQL. A sample figure is given as follows:

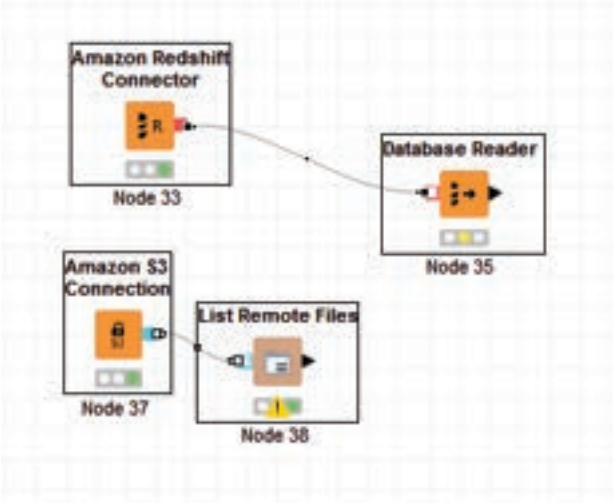


Apache Spark integrated with Node Workflow

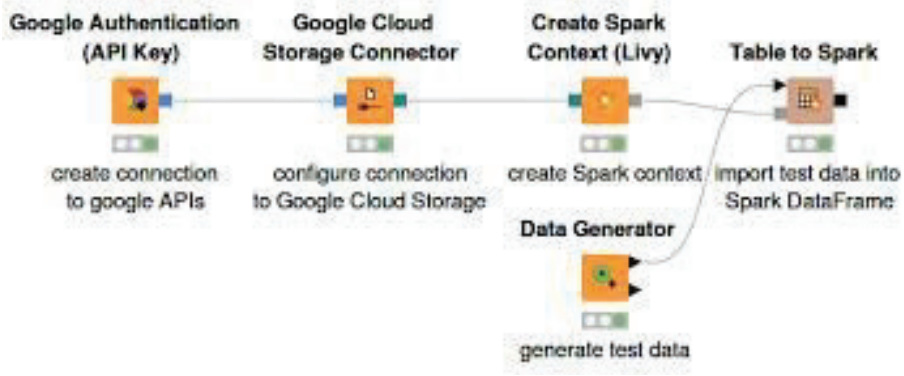
3 **Apache Hive:** KNIME can connect to Apache Hive, a data warehousing and SQL-like query language tool for Hadoop. You can use the "Hive Connector" and "Hive Table Selector" nodes to interact with Hive tables and data stored in Hive.

4 **Big Data Databases:** KNIME provides connectors for various Big Data databases, such as HBase, Cassandra, and more. These nodes enable you to query and process data stored in these systems directly from KNIME.

5 **Other Sources:** KNIME also supports connecting to other data sources, including cloud-based Big Data platforms like Amazon S3, Google Cloud Storage, and Azure Blob Storage, as well as web services and APIs. A sample figure is given as follows:



Amazon S3 Connection node

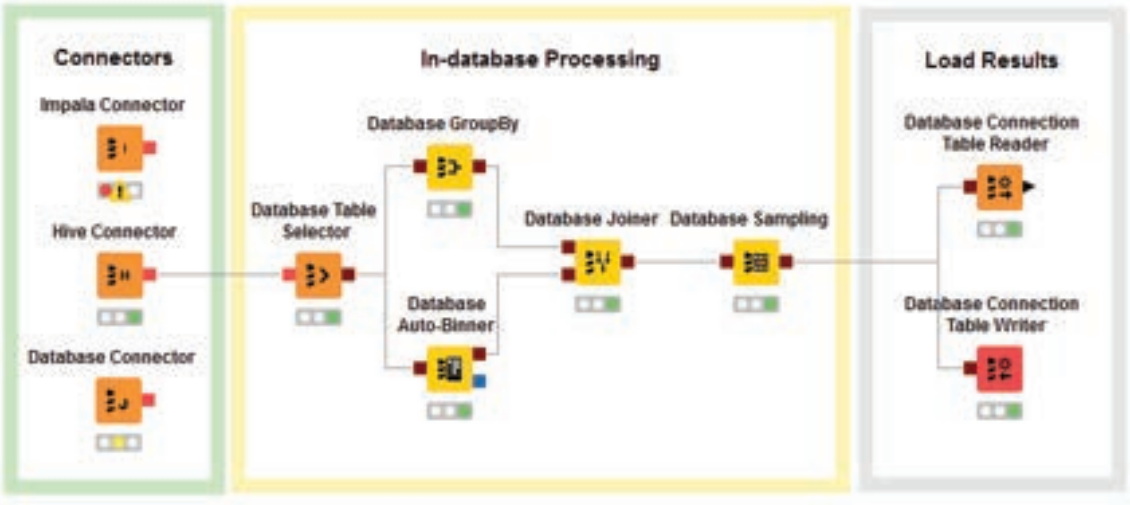


Google Cloud Storage Connectors with Workflow

4.2 HANDLING BIG DATA IN KNIME

Handling Big Data efficiently within KNIME involves strategies to manage large volumes of data during data preprocessing, analysis, and transformation:

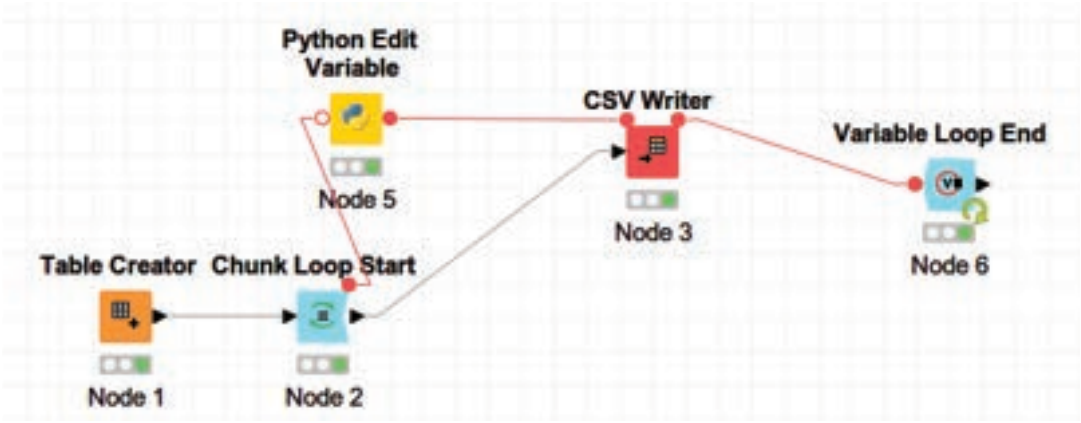
1 **Data Sampling:** When working with Big Data, it's often impractical to process the entire dataset at once. You can use KNIME's sampling capabilities to work with a representative subset of the data for initial exploration and modeling. A sample figure is given as follows:



Big Data's Data Sampling Workflow

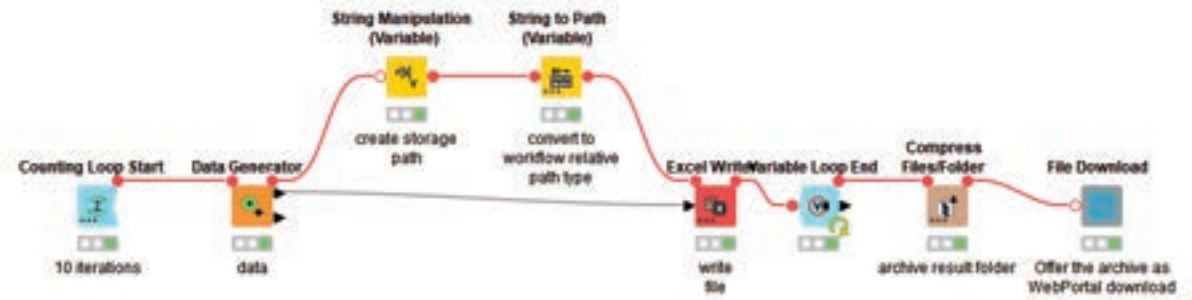
2 **Distributed Computing:** KNIME leverages distributed computing frameworks like Apache Spark and Hadoop to process large datasets in parallel. This parallel processing improves performance by utilizing multiple processing nodes or cores.

3 **Data Chunking:** To prevent memory constraints, KNIME can process data in smaller, manageable chunks. This approach allows you to analyse and transform data that doesn't fit entirely in memory. A sample figure is given as follows:



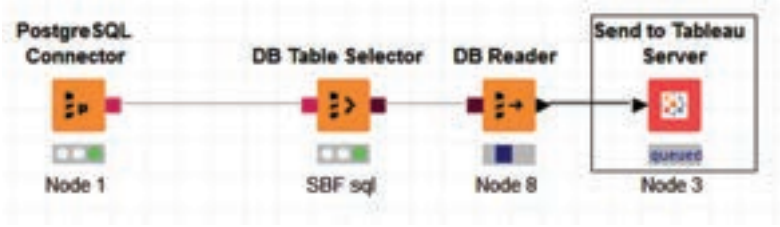
Data Chunking node examples

4 **Data Compression:** Data compression techniques can be employed to reduce storage requirements and optimize data transfer between nodes and across the network. A sample figure is given as follows:



Data Compression node with Workflow

5 **In-Database Processing:** KNIME supports in-database processing, where data remains within the database for analysis. This minimizes data movement and enhances performance. A sample figure is given as follows:

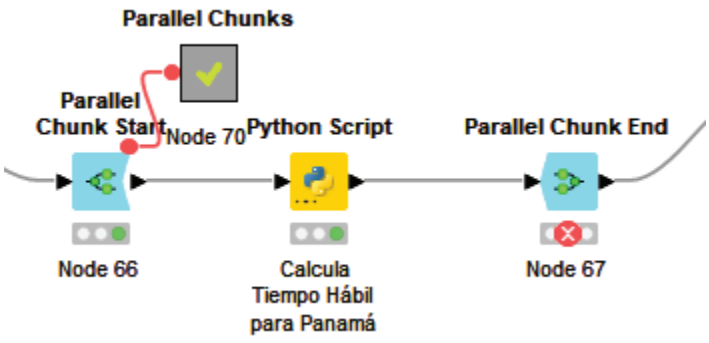


In Database processing example

4.3 DISTRIBUTED DATA PROCESSING

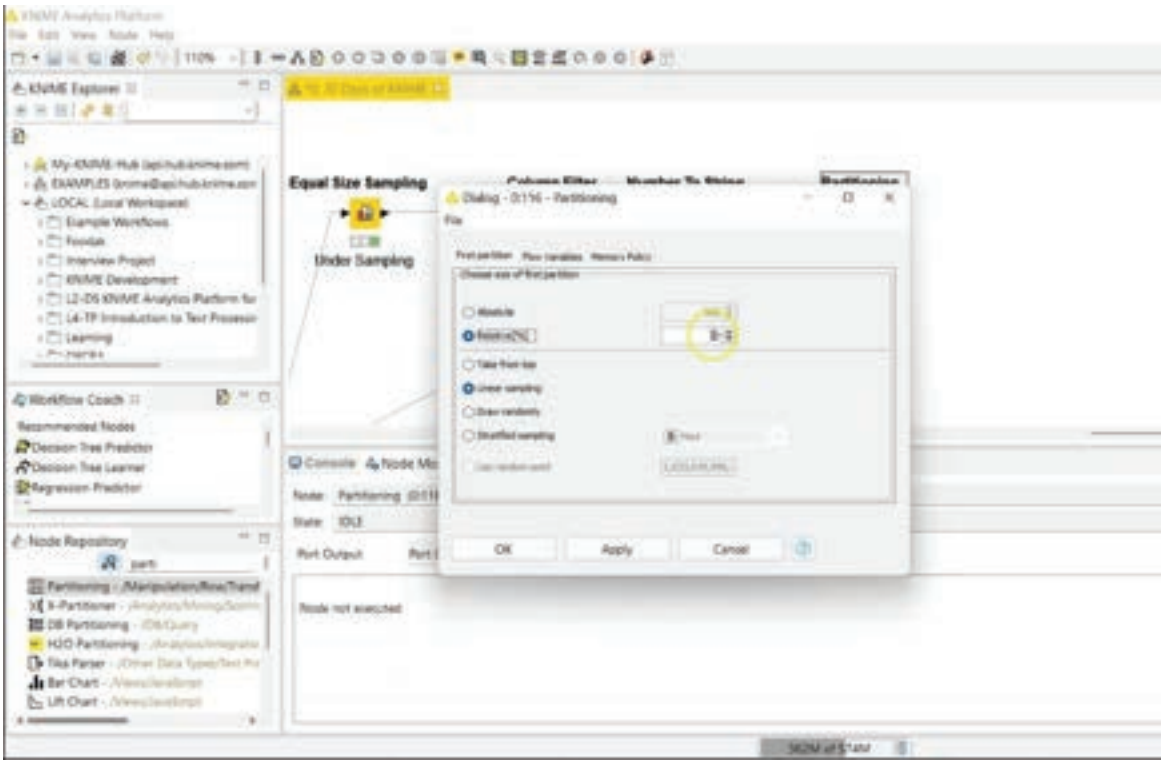
Distributed data processing is a key capability in KNIME for efficiently analysing and processing Big Data:

1 **Parallel Execution:** KNIME uses parallel execution to distribute data processing tasks across multiple nodes or worker machines. This parallelization significantly improves performance and reduces processing time. A sample figure is given as follows:



Parallel Execution example

2 **Data Partitioning:** KNIME automatically partitions data into smaller chunks that can be processed in parallel by different nodes or worker nodes. This ensures efficient resource utilization. A sample figure is given as follows:



Partitioning configuration

3 **Load Balancing:** KNIME's distributed processing capabilities ensure load balancing, distributing tasks evenly across available resources to maximize their utilization.

4 **Scalability:** KNIME can scale horizontally by adding more compute resources to handle larger datasets and more complex analyses.

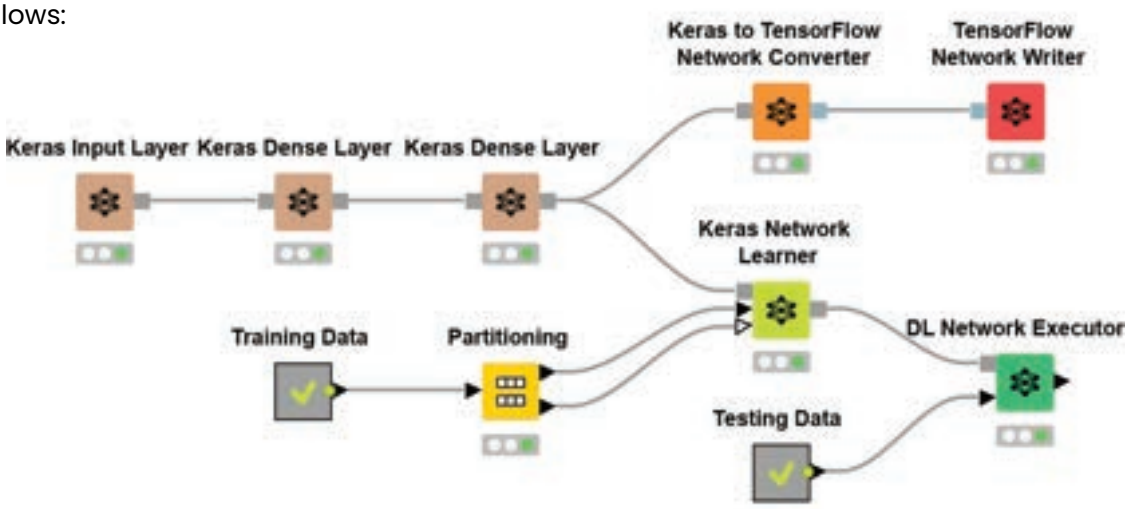
4.4 BIG DATA ANALYTICS WITH KNIME

Once you've connected to Big Data sources, set up distributed data processing, and prepared your data, you can perform various Big Data analytics tasks using KNIME:

1 **Data Exploration:** Use KNIME's data exploration tools to gain insights into your Big Data, generate summary statistics, and create visualizations.

2 **Data Preprocessing:** Clean, transform, and prepare Big Data for analysis using KNIME's data preprocessing nodes. This includes handling missing values, encoding categorical variables, and more

3 **Machine Learning:** KNIME offers a wide range of machine learning algorithms that can be applied to Big Data, including classification, regression, clustering, and anomaly detection. You can build and evaluate models for predictive analytics. A sample figure is given as follows:

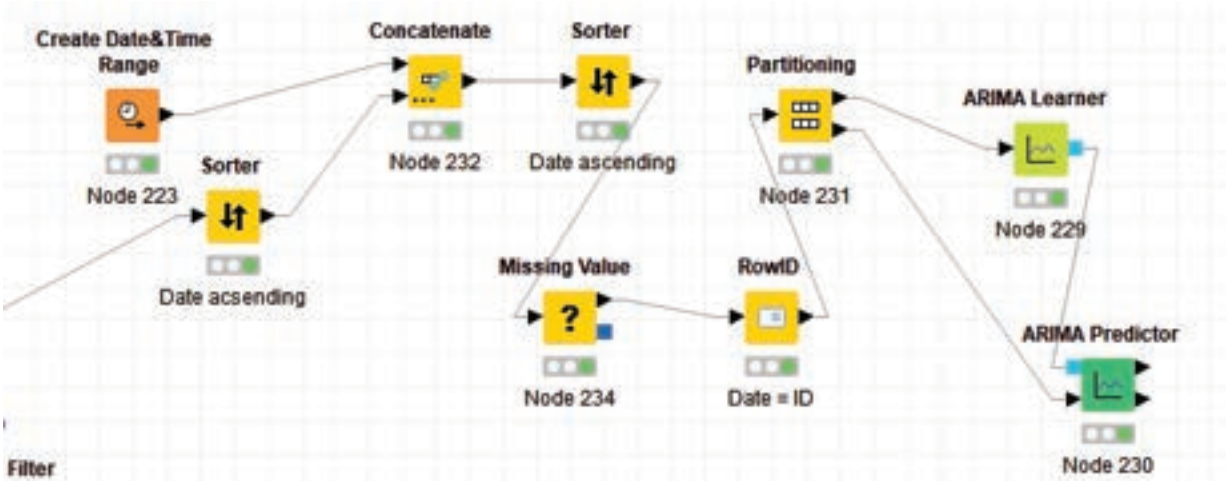


Machine Learning Workflow example

4 **Text and Image Analytics:** KNIME supports text mining and image analysis, allowing you to extract valuable information from unstructured Big Data sources. A sample figure is given as follows:

Text & Image Analytics nodes example

5 **Advanced Analytics:** Conduct advanced analytics, such as time series analysis, network analysis, and geospatial analysis, on your Big Data. A sample figure is given as follows:



Advance Analytics Workflow example

6 **Data Visualization and Reporting:** Create visualizations and reports to communicate your findings effectively. KNIME provides various visualization nodes for generating charts and graphs.

7 **Deployment and Integration:** Deploy your Big Data analytics workflows for production use. Export models and predictions for integration with other applications and data pipelines.

8 **Monitoring and Maintenance:** Continuously monitor and maintain your Big Data analytics workflows. Optimize workflows for performance and keep KNIME and its extensions up to date.

9 **Collaboration and Sharing:** Collaborate with team members and stakeholders by sharing KNIME workflows, reports, and results. Use KNIME Server for collaborative work and scheduled execution.

KNIME's flexibility, scalability, and integration with Big Data technologies make it a valuable tool for data scientists and analysts working with large and complex datasets. With the right configurations and nodes, KNIME can efficiently handle Big Data analytics tasks and provide valuable insights.

EXERCISE

Connecting to Big Data Sources:

- Launch KNIME Analytics Platform.
- Create a new workflow in the Workflow Editor.
- Use the Node Repository to add a "Big Data Connect" node to your workflow.
- Configure the "Big Data Connect" node to establish a connection to a Big Data source of your choice (e.g., Hadoop, Spark, or a database).
- Ensure you have the necessary credentials and connection details.
- Execute the workflow to establish the connection.

Handling Big Data in KNIME:

- Add a "Data Explorer" node to your workflow.
- Connect the "Big Data Connect" node to the "Data Explorer" node.
- Configure the "Data Explorer" node to retrieve a sample subset of Big Data for exploration.
- Execute the workflow to view and explore the data in KNIME.

Distributed Data Processing:

- Add a "Spark Executor" node to your workflow.
- Connect the "Big Data Connect" node to the "Spark Executor" node.
- Configure the "Spark Executor" node to perform a distributed data processing task (e.g., a simple data transformation or aggregation).
- Execute the workflow to apply distributed data processing using Spark.

Big Data Analytics with KNIME:

- Add a "Machine Learning" node (e.g., "Random Forest") to your workflow.
- Connect the "Spark Executor" node to the "Machine Learning" node.
- Configure the machine learning node to build a predictive model using the Big Data.
- Execute the workflow to perform Big Data analytics, such as predictive modeling.

Validation and Reflection:

- Review the results of the data exploration, distributed data processing, and analytics tasks.
- Reflect on your experience with working on Big Data in KNIME.
- Note any challenges faced and how you overcame them.
- Consider potential real-world applications where KNIME's Big Data capabilities could be valuable.

This exercise will provide you with hands-on experience in connecting to Big Data sources, handling Big Data within KNIME, performing distributed data processing, and conducting advanced Big Data analytics, demonstrating the platform's capabilities in working with large-scale datasets.`

SUMMARY

In conclusion, navigating the realm of Big Data is a critical endeavour in today's data-driven landscape. The ability to effectively connect to diverse Big Data sources equips data professionals with the means to tap into vast repositories of information, unlocking valuable insights. Furthermore, mastering the art of handling Big Data within KNIME ensures that data processing remains efficient and scalable, even as datasets expand exponentially. Understanding distributed data processing principles empowers individuals to harness the full potential of distributed computing, making it possible to analyse, model, and extract meaningful patterns from massive datasets. Finally, by leveraging KNIME for Big Data analytics, organizations and data scientists can turn this wealth of information into actionable knowledge, driving innovation and informed decision-making across various domains and industries. In this era of Big Data, these skills and insights are invaluable for staying competitive and relevant in the data analytics landscape.

REFERENCE

1 *https://www.knime.com/*
2 *https://www.knime.com/getting-started-guide*
3 *https://www.knime.com/knimepress*
4 *KNIME Beginner's Luck Book by Rosario_Silipo*
5 *KNIME Advanced Luck A Guide to KNIME Analytics Platform for Advanced Users by Rosario Silipo, Jeanette Prinz*

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 What does the term "Big Data" generally refer to?
- a) Small and manageable datasets
- b) Extremely large and complex datasets
- c) Structured data only
- d) Data from a single source

2 In KNIME, what does "connecting to Big Data sources" primarily involve?
- a) Creating animations
- b) Importing data from spreadsheets
- c) Establishing connections to large-scale data systems
- d) Printing documents

3 Why is it important to connect to Big Data sources in KNIME?
- a) To generate random data
- b) To create 3D models
- c) To access and analyse massive datasets
- d) To change node colors

4 What does "handling Big Data in KNIME" refer to?
- a) Cleaning data
- b) Working with small datasets
- c) Efficiently processing and analysing large-scale data
- d) Playing music files

5 What is one benefit of efficiently handling Big Data in KNIME?
- a) Slower data processing
- b) Increased hardware requirements
- c) Improved performance and resource utilization
- d) Reduced access to data

6 What is the primary goal of distributed data processing?
- a) To make data analysis easier
- b) To handle data from a single source
- c) To process data on a single machine
- d) To improve scalability and parallelism

7 In the context of distributed data processing, what does "parallelization" mean?
- a) Running tasks sequentially
- b) Running tasks concurrently on multiple machines or processors
- c) Ignoring data
- d) Transforming data into images

8 How can KNIME facilitate Big Data analytics?
   a) By limiting data access
   b) By restricting data transformations
   c) By providing tools for data exploration, modeling, and analysis
   d) By sending emails

9 What is a common step in Big Data analytics with KNIME?
   a) Creating visual designs
   b) Modeling data
   c) Adding unnecessary complexity
   d) Deleting data

10 What is the primary focus of Big Data analytics using KNIME?
   a) Generating random data
   b) Extracting insights and patterns from large and complex datasets
   c) Changing node colors
   d) Playing video games

11 Which of the following is NOT a benefit of connecting to Big Data sources in KNIME?
   a) Access to massive datasets
   b) Improved data quality
   c) Enhanced data analysis capabilities
   d) Faster computer processing speed

12 In KNIME, how does efficient handling of Big Data impact workflow performance?
   a) It slows down data processing.
   b) It increases the complexity of the workflow.
   c) It improves performance by optimizing resource usage.
   d) It has no effect on workflow performance.

13 What is a key advantage of distributed data processing in KNIME?
   a) It simplifies data analysis.
   b) It reduces the need for powerful hardware.
   c) It enhances scalability for processing large datasets.
   d) It makes data inaccessible.

14 Which of the following is NOT a typical step in Big Data analytics with KNIME?
   a) Data exploration
   b) Parallelization of tasks
   c) Data modeling
   d) Sending emails

15 What does "resource utilization" refer to in the context of Big Data handling in KNIME?
   a) The process of collecting data
   b) The efficient use of computing resources like CPU and memory
   c) The creation of 3D models
   d) The management of data sources

Answers

1 b) *Extremely large and complex datasets*
2 c) *Establishing connections to large-scale data systems*
3 c) *To access and analyse massive datasets*
4 c) *Efficiently processing and analysing large-scale data*
5 c) *Improved performance and resource utilization*
6 d) *To improve scalability and parallelism*
7 b) *Running tasks concurrently on multiple machines or processors*
8 c) *By providing tools for data exploration, modeling, and analysis*
9 b) *Modeling data*
10 b) *Extracting insights and patterns from large and complex datasets*
11 d) *Faster computer processing speed*
12 c) *It improves performance by optimizing resource usage.*
13 c) *It enhances scalability for processing large datasets.*
14 d) *Sending emails*
15 b) *The efficient use of computing resources like CPU and memory*

CHAPTER 5
TRANSFORMATION, ANALYSIS & DATA MINING

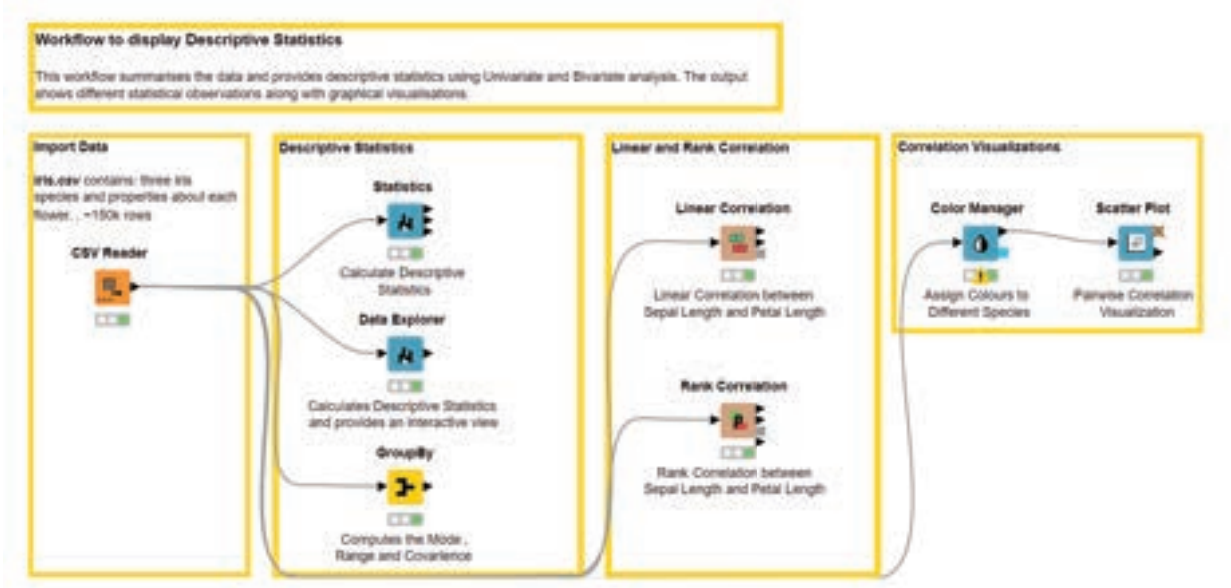
LEARNING OBJECTIVES

- Master data transformation and manipulation techniques in KNIME to prepare raw data for analysis effectively.
- Develop proficiency in performing statistical analysis within KNIME, including descriptive and inferential statistics.
- Learn to create machine learning workflows for predictive analytics and explore various algorithms for classification, regression, and clustering.
- Understand the principles of predictive analytics and data mining, enabling the extraction of valuable patterns and insights from data to support decision-making processes.

5.1 STATISTICAL ANALYSIS IN KNIME

KNIME supports various statistical analysis tasks:

1. **Descriptive Statistics:** Use nodes like "Statistics" and "Data Explorer" to compute descriptive statistics, including measures of central tendency, dispersion, and frequency distributions.
2. **Hypothesis Testing:** Conduct hypothesis testing with nodes like "Group Comparison" for comparing means, "Chi-Square Test" for categorical data, and others.
3. **Correlation Analysis:** Determine relationships between variables using correlation analysis nodes.
4. **ANOVA and Regression Analysis:** Perform analysis of variance (ANOVA) and regression analysis to explore relationships between dependent and independent variables.
5. **Time Series Analysis:** Analyse time series data with specialized nodes for forecasting, trend analysis, and seasonal decomposition. A sample figure is given as follows:



Statistical Analysis related Workflow example

5.2 MACHINE LEARNING WORKFLOWS

KNIME offers extensive support for creating and executing machine learning workflows:

1 **Model Selection:** Use nodes for model selection, such as "Variable Selection" and "Feature Selection," to identify the most relevant features for modeling.

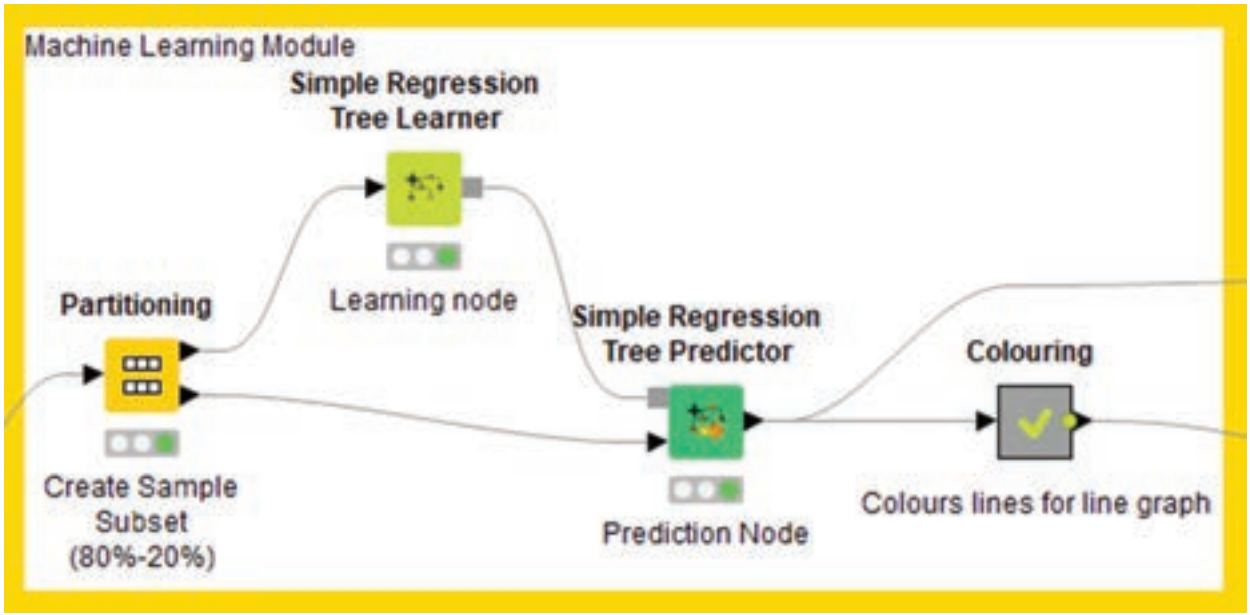
2 **Model Training:** Train machine learning models using nodes for various algorithms, including decision trees, random forests, support vector machines, and neural networks.

3 **Model Evaluation:** Evaluate models with nodes like "Scorer" to assess their performance using metrics like accuracy, precision, recall, F1-score, and ROC curves.

4 **Cross-Validation:** Implement cross-validation techniques using nodes like "Cross-Validation Loop" to assess model generalization.

5 **Ensemble Learning:** Build ensemble models using nodes like "Ensemble Learner" to combine multiple models for improved predictive performance.

6 **Hyperparameter Tuning:** Optimize model hyperparameters with nodes like "Parameter Optimization" to achieve the best model performance.
A sample figure is given as follows:

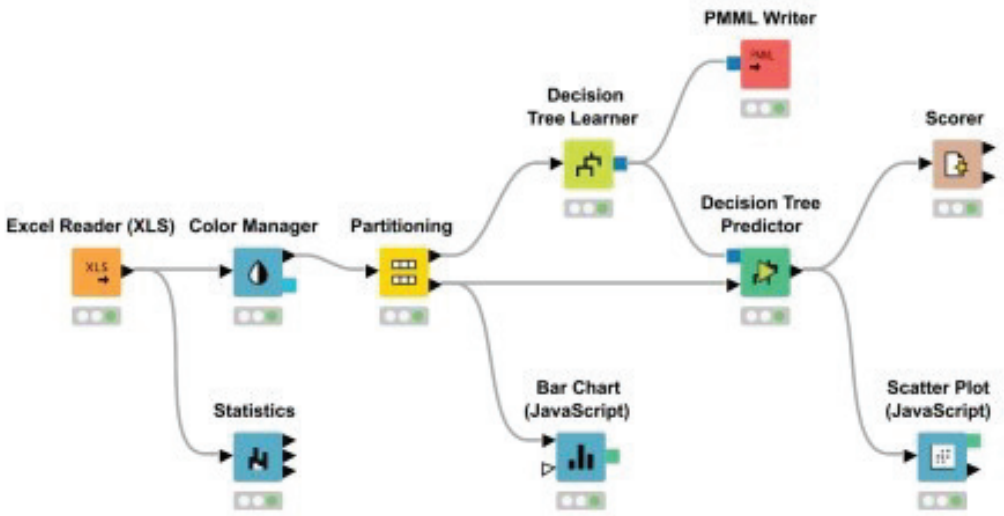


Machine Learning related node and workflow example

5.3 PREDICTIVE ANALYTICS AND DATA MINING

KNIME is a powerful tool for predictive analytics and data mining:

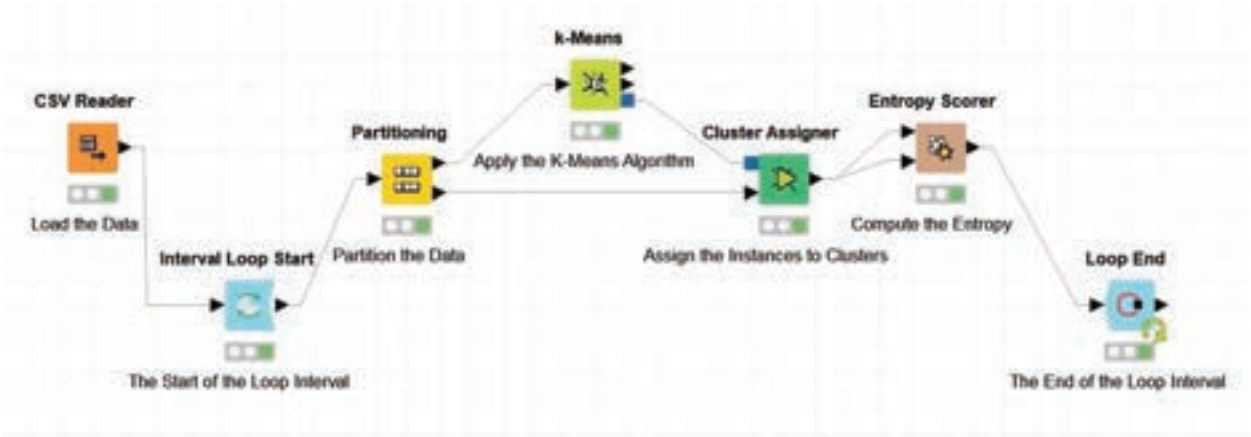
1 **Classification:** Build classification models to predict categorical outcomes. KNIME supports various algorithms like logistic regression, decision trees, and k-nearest neighbours. A sample figure is given as follows:



Classification models to predict categorical outcomes Workflow example

2 **Regression:** Perform regression analysis to predict numeric outcomes. KNIME includes regression algorithms like linear regression and support vector regression.

3 **Clustering:** Use clustering algorithms to group similar data points together. KNIME offers k-means clustering, hierarchical clustering, and DBSCAN, among others. A sample figure is given as follows:



K-means Clustering workflow example

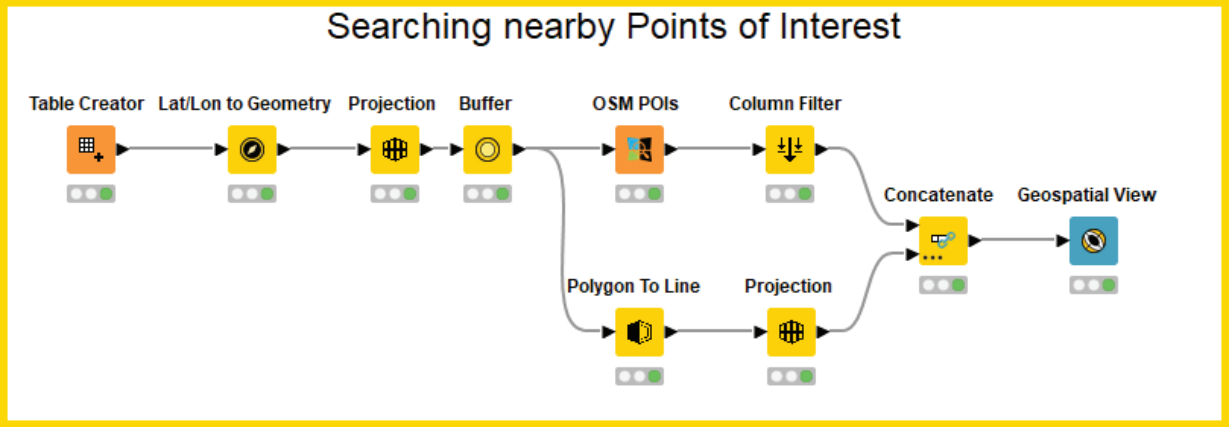
4 **Anomaly Detection:** Detect anomalies or outliers in data using specialized nodes like "Local Outlier Factor."

5 **Association Rule Mining:** Discover patterns and associations in data using association rule mining nodes.

6 **Time Series Forecasting:** Forecast future values of time series data using dedicated nodes for time series analysis and prediction.

7 **Text Mining and NLP:** KNIME provides nodes for text mining and natural language processing (NLP), allowing you to analyse and extract insights from unstructured text data.

8 **Geospatial Analysis:** Perform geospatial analysis and visualization using nodes for geographic information system (GIS) data. A sample figure is given as follows:



Geographic information system (GIS) data workflow example

KNIME's versatility, extensive library of nodes, and user-friendly visual interface make it a valuable tool for data scientists and analysts involved in data transformation, statistical analysis, machine learning, and predictive analytics.

EXERCISE

Data Transformation and Manipulation:

- Launch KNIME Analytics Platform.
- Create a new workflow in the Workflow Editor.
- Use the Node Repository to add a "Data Manipulation" node to your workflow.
- Load a sample dataset (e.g., CSV or Excel) into the workflow using a "File Reader" node.
- Configure the "Data Manipulation" node to perform a transformation task, such as filtering rows, renaming columns, or aggregating data.
- Execute the workflow to see the transformed data.

Statistical Analysis in KNIME:

- Add a "Descriptive Statistics" node to your workflow.
- Connect the "Data Manipulation" node to the "Descriptive Statistics" node.
- Configure the node to calculate descriptive statistics for specific columns in the dataset.
- Execute the workflow to obtain statistical summaries.

Machine Learning Workflows:

- Add a "Decision Tree" learner node to your workflow.
- Connect the "Data Manipulation" node to the "Decision Tree" learner node.
- Add a "Decision Tree Predictor" node to your workflow and connect it to the learner node.
- Configure the nodes to build and apply a decision tree model.
- Execute the workflow to train and test the model.

Predictive Analytics and Data Mining:

- Add a "Data Mining" node (e.g., "Association Rule Learner") to your workflow.
- Connect the "Data Manipulation" node to the data mining node.
- Configure the node to discover patterns or associations in the data.
- Execute the workflow to perform data mining.

Validation and Reflection:

- Review the results of the data transformation, statistical analysis, machine learning, and data mining tasks.
- Reflect on your experience with KNIME for data transformation and analysis.
- Note any insights or patterns you discovered during the exercise.
- Consider how these techniques can be applied to real-world data analysis scenarios.

This exercise will provide you with hands-on experience in using KNIME to transform, analyse, and mine data, making it a valuable tool for data professionals and analysts.

SUMMARY

In conclusion, the journey through the intricacies of Transformation, Analysis, and Data Mining within the KNIME environment equips individuals and organizations with the tools and insights necessary to derive value from their data. Proficiency in data transformation and manipulation ensures that raw data is refined into a usable format, laying the foundation for meaningful analysis. The ability to conduct statistical analysis empowers data professionals to uncover patterns and trends within their datasets, fostering data-driven decision-making. Machine learning workflows provide the means to leverage advanced algorithms and techniques for predictive analytics, enabling the anticipation of future outcomes and trends. Moreover, the discipline of predictive analytics and data mining enables the extraction of hidden gems from vast datasets, transforming information into actionable knowledge. These capabilities collectively contribute to informed decision-making, innovation, and competitive advantage in today's data-centric landscape, making them indispensable skills for data professionals and organizations alike.

REFERENCE

1 *https://www.knime.com/*
2 *https://www.knime.com/getting-started-guide*
3 *https://www.knime.com/knimepress*
4 *KNIME Beginner's Luck Book by Rosario_Silipo*
5 *KNIME Advanced Luck A Guide to KNIME Analytics Platform for Advanced Users by Rosario Silipo, Jeanette Prinz*

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 What is the primary purpose of data transformation and manipulation in KNIME?
   a) To create visual designs
   b) To prepare raw data for analysis
   c) To generate random data
   d) To print documents

2 In KNIME, what do statistical analysis techniques help users accomplish?
   a) Play video games
   b) Explore data patterns and trends
   c) Design 3D animations
   d) Edit images

3 Which of the following is NOT a common statistical analysis task in KNIME?
   a) Descriptive statistics
   b) Inferential statistics
   c) Creating websites
   d) Hypothesis testing

4 What is the primary purpose of machine learning workflows in KNIME?
   a) To send emails
   b) To create data visualizations
   c) To predict future outcomes based on historical data
   d) To browse the internet

5 In the context of machine learning, what does "classification" refer to?
   a) Arranging data in alphabetical order
   b) Grouping data into categories or classes
   c) Deleting data
   d) Changing node colors

6 What is the primary goal of predictive analytics in KNIME?
   a) To create 3D models
   b) To perform data transformations
   c) To extract valuable insights and patterns from data
   d) To generate random data points

6 What is the primary goal of predictive analytics in KNIME?
   a) To create 3D models
   b) To perform data transformations
   c) To extract valuable insights and patterns from data
   d) To generate random data points

7 What is data mining in KNIME?
   a) The process of browsing websites
   b) The process of transforming data into images
   c) The process of extracting hidden knowledge and patterns from data
   d) The process of sending emails

8 Which type of statistical analysis helps in summarizing and describing data characteristics?
   a) Descriptive statistics
   b) Inferential statistics
   c) Data visualization
   d) Data manipulation

9 Which part of KNIME is typically used for building machine learning workflows?
   a) Spreadsheet editor
   b) Workflow editor
   c) Image editor
   d) Web browser

10 In machine learning, what does "regression" refer to?
   a) Predicting categories or classes
   b) Predicting numerical values or quantities
   c) Predicting data transformations
   d) Predicting data visualization patterns

11 What is a common technique in predictive analytics to handle imbalanced datasets?
   a) Deleting data
   b) Oversampling the minority class
   c) Changing font size
   d) Creating animations

12 Which of the following is NOT a common step in machine learning workflows?
   a) Data preprocessing
   b) Model training and evaluation
   c) Statistical analysis
   d) Predictive modeling

13 What is clustering in the context of data mining?
   a) Deleting data
   b) Grouping data points into similar clusters or groups
   c) Sending emails
   d) Creating 3D animations

14 What is an essential outcome of predictive analytics in KNIME?
   a) Generating random data
   b) Extracting valuable patterns and insights from data
   c) Editing images
   d) Creating websites

15 Which of the following is NOT a common machine learning algorithm used in KNIME?
   a) Decision trees
   b) K-means clustering
   c) Web scraping
   d) Support vector machines

Answers

1 *b) To prepare raw data for analysis*
2 *b) Explore data patterns and trends*
3 *c) Creating websites*
4 *c) To predict future outcomes based on historical data*
5 *b) Grouping data into categories or classes*
6 *c) To extract valuable insights and patterns from data*
7 *c) The process of extracting hidden knowledge and patterns from data*
8 *a) Descriptive statistics*
9 *b) Workflow editor*
10 *b) Predicting numerical values or quantities*
11 *b) Oversampling the minority class*
12 *c) Statistical analysis*
13 *b) Grouping data points into similar clusters or groups*
14 *b) Extracting valuable patterns and insights from data*
15 *c) Web scraping*

CHAPTER 6
TRANSFORMATION, ANALYSIS & DATA MINING

LEARNING OBJECTIVES

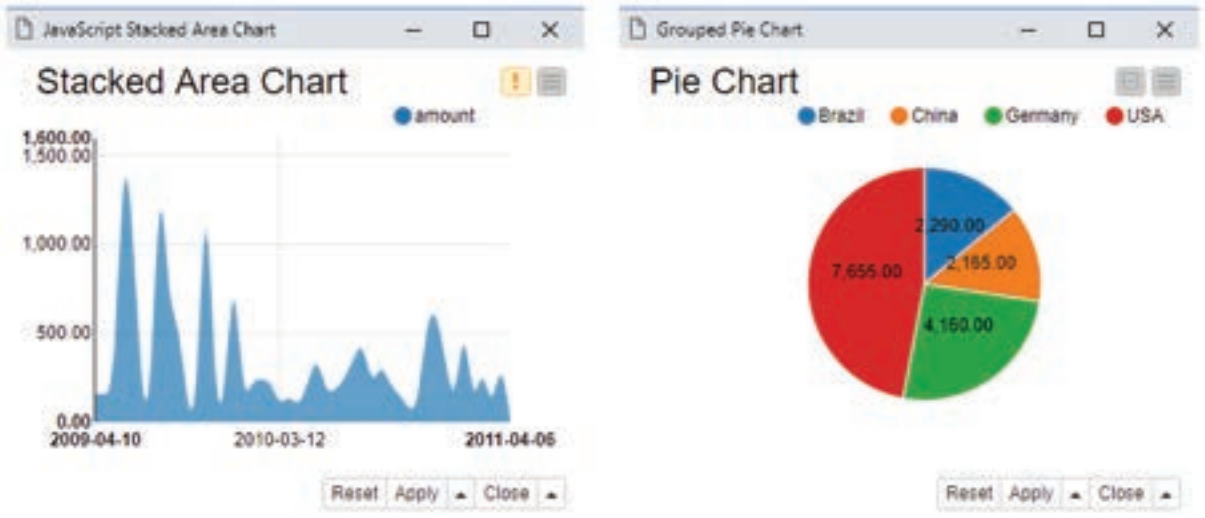
- Master the art of data visualization in KNIME, enabling the creation of informative and visually appealing charts, graphs, and plots to convey insights effectively.
- Learn to build interactive dashboards within KNIME, allowing users to explore data dynamically and gain deeper insights through interactive elements.
- Understand the process of model deployment and integration, enabling the seamless integration of machine learning models into production environments for real-world applications.
- Explore automation and reporting capabilities in KNIME, facilitating the automated execution of workflows and the generation of reports for streamlined data analysis and decision support.

6.1 DATA VISUALIZATION IN KNIME

Data visualization is a powerful way to communicate insights and patterns in your data. In KNIME, you can create compelling visualizations to aid in data exploration and analysis:

- **Visualization Nodes:** KNIME offers a variety of nodes for creating visualizations, including scatter plots, bar charts, line charts, heatmaps, and more. You can configure these nodes to represent your data effectively.

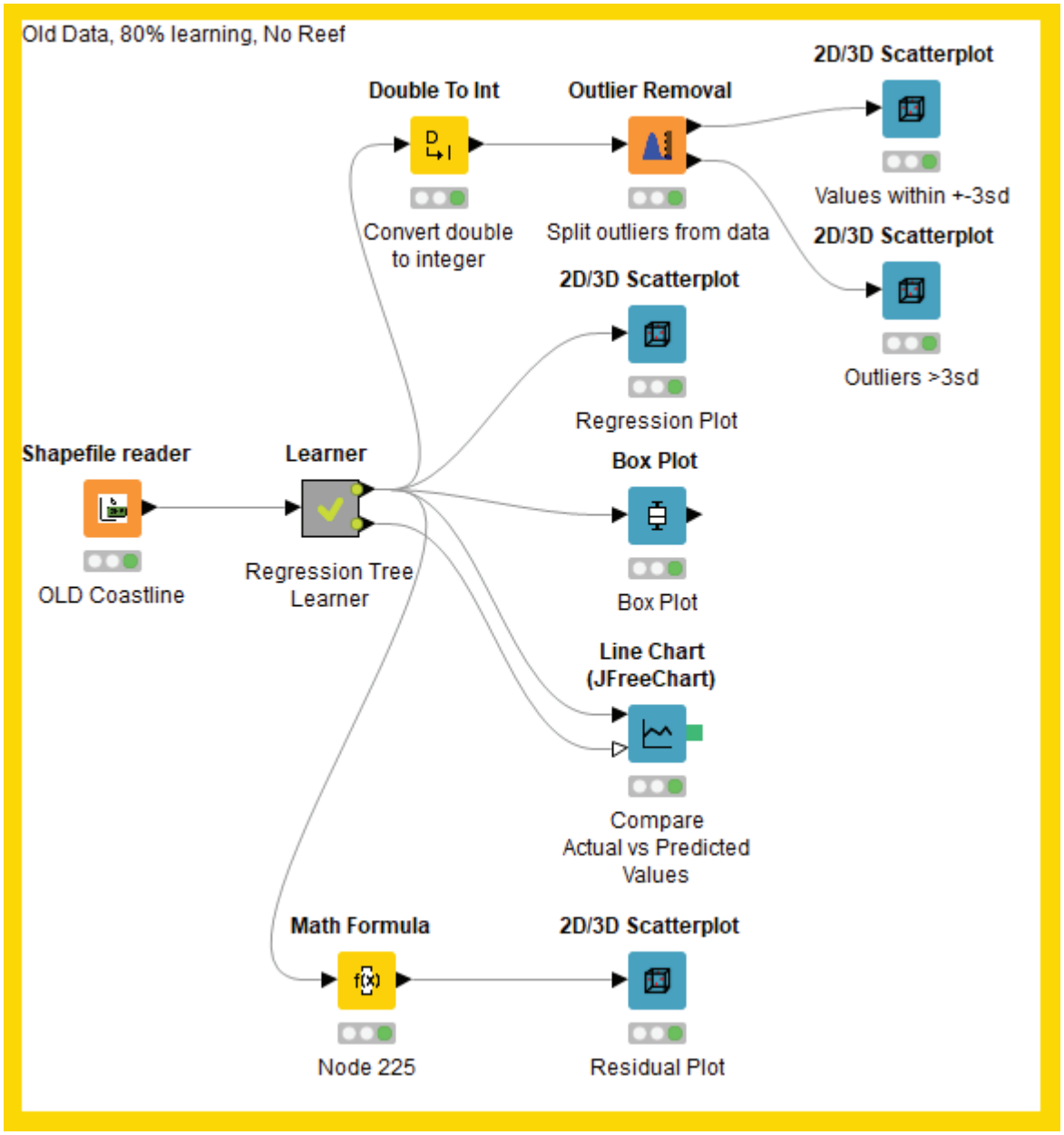
- **Interactive Plots:** KNIME supports interactive plots that allow users to explore data dynamically. Interactive features include zooming, panning, tooltips, and filtering. A sample figure is given as follows:



Interactive Plots Output example

- **Customization:** Customize the appearance of your visualizations, such as color schemes, labels, and axis scaling, to ensure clarity and relevance.

- **Automated Visualization:** KNIME can automate the generation of visualizations using data-driven approaches. For example, you can create a series of plots for multiple columns or generate visualizations based on certain conditions. A sample figure is given as follows:

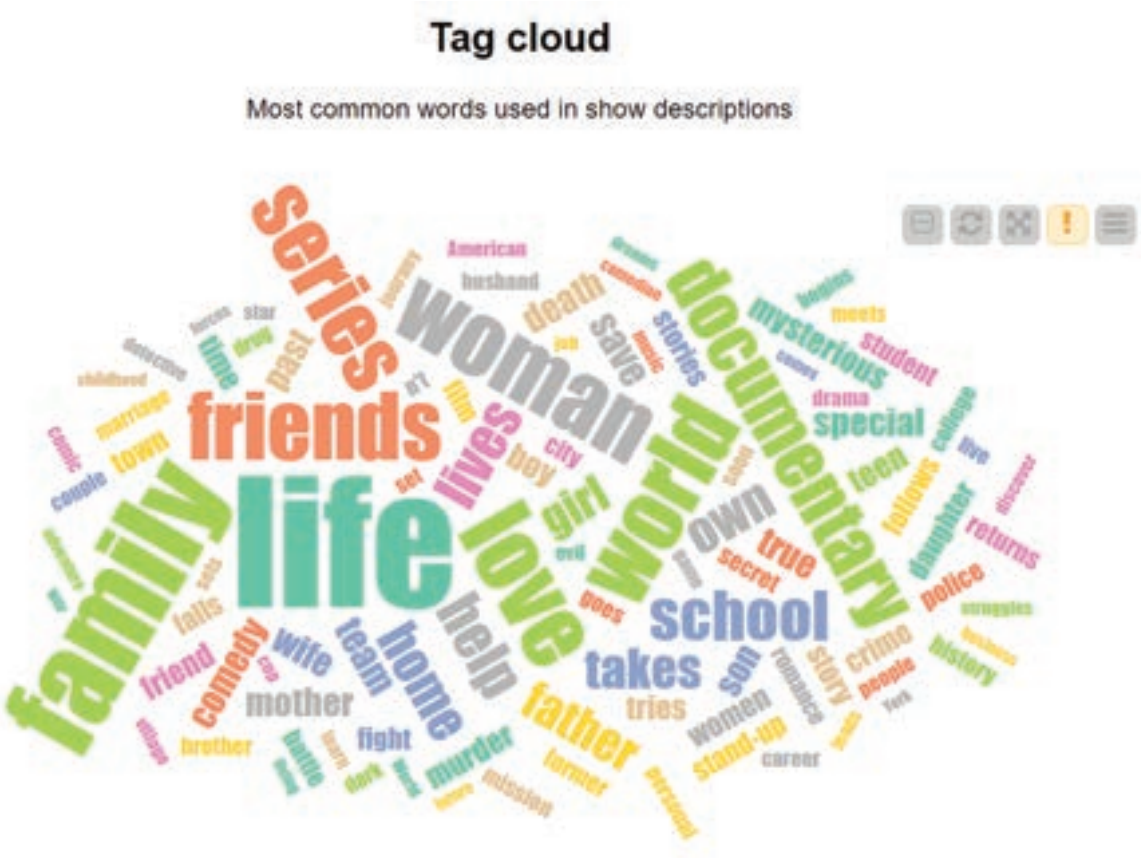


Automated Visualization Workflow example

6.2 CREATING INTERACTIVE DASHBOARDS

Interactive dashboards provide a dynamic way to present data and allow users to interact with it. In KNIME, you can build interactive dashboards to convey insights effectively:

- **Dashboard Components:** KNIME offers a range of dashboard components like tables, charts, filters, and input widgets that can be combined to create interactive dashboards.
- **Drag-and-Drop Design:** Build dashboards using a user-friendly, drag-and-drop interface. You can arrange components and link them to control each other dynamically.
- **Real-Time Updates:** Dashboards can be designed to update in real-time as data changes, enabling users to see the latest information instantly.
- **Parameterization:** Create dashboards with parameterized inputs, allowing users to customize views based on their preferences or specific analysis requirements. A sample figure is given as follows:

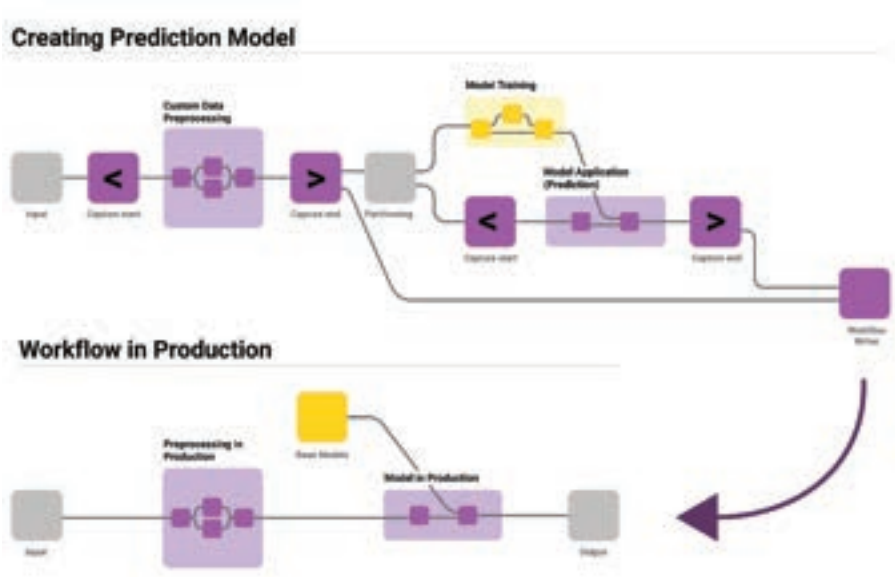


Interactive Output example

6.3 MODEL DEPLOYMENT AND INTEGRATION

Deploying machine learning models and integrating them into production systems is crucial for making data-driven decisions. KNIME supports model deployment and integration:

- **Export Models:** KNIME allows you to export trained machine learning models in standard formats (e.g., PMML) for deployment in various environments, such as web applications or databases.
- **RESTful Web Services:** You can deploy models as RESTful web services using KNIME Server. This enables real-time predictions and integration with other applications.
- **Batch Processing:** Automate model deployment by integrating KNIME workflows into batch processing pipelines to generate predictions on new data regularly.
- **Database Integration:** Integrate models with databases to perform in-database scoring, making predictions directly within the database engine. A sample figure is given as follows:

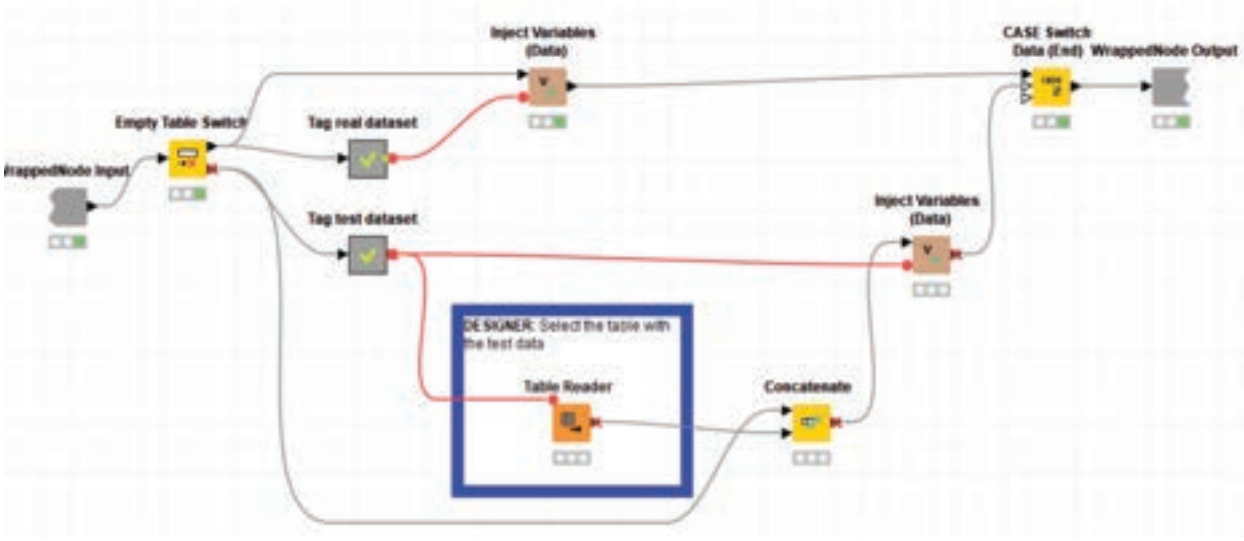


Model Deployment and Integration workflow template

6.4 AUTOMATION AND REPORTING IN KNIME

Automation and reporting are essential for streamlining workflows and sharing insights. KNIME offers automation and reporting features:

- **Workflow Automation:** Automate repetitive tasks and data processing steps using KNIME's workflow automation capabilities. Schedule workflows to run at specific times or events. A sample figure is given as follows:



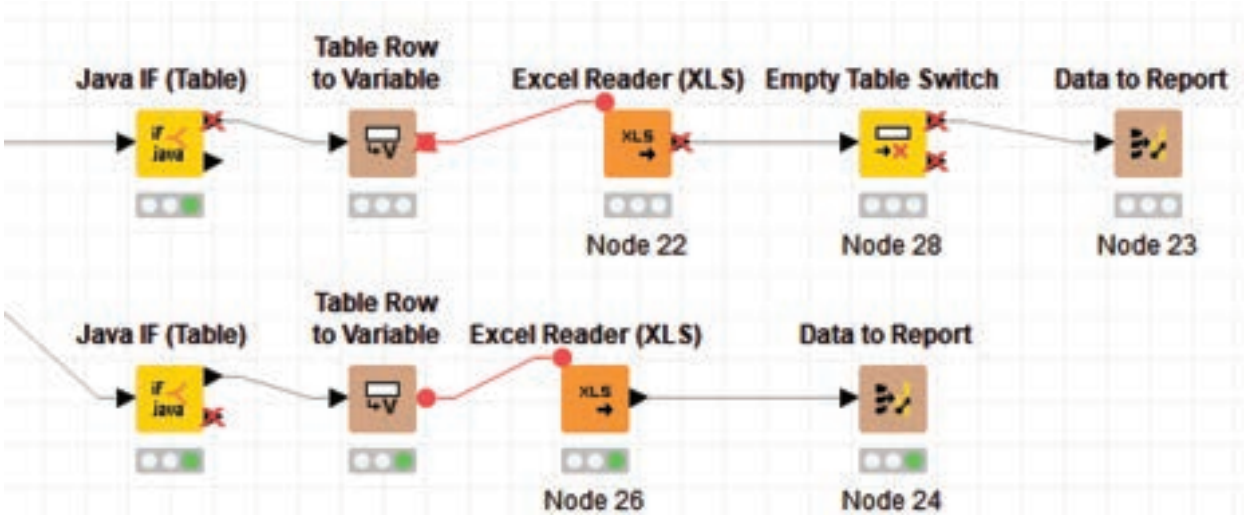
Workflow automation example

- **Report Generation:** Create customizable reports in KNIME with text, tables, charts, and visualizations. Reports can be generated automatically as part of a workflow or on-demand.

- **Data Export:** Export data, results, and reports in various formats, including PDF, Excel, CSV, and more, to share insights with stakeholders.

- **Integration with External Systems:** KNIME can integrate with external systems and databases to import data, trigger workflows, and export results seamlessly.

- **Notifications:** Configure notifications and alerts to inform users or administrators about workflow status, errors, or specific events. A sample figure is given as follows:



Configure notifications and alerts template

KNIME's capabilities in data visualization, interactive dashboards, model deployment, automation, and reporting make it a comprehensive platform for end-to-end data analytics and decision support, from data exploration to sharing insights with stakeholders.

EXERCISE

Data Visualization in KNIME:

- Launch KNIME Analytics Platform.
- Create a new workflow in the Workflow Editor.
- Add a "Data Reader" node to the workflow and load a sample dataset (e.g., a CSV file).
- Use the Node Repository to add a "Scatter Plot" node to visualize the data.
- Customize the plot by selecting appropriate columns for X and Y axes.
- Execute the workflow to generate and view the scatter plot.

Creating Interactive Dashboards:

- Add an "Interactive Table" node to your workflow.
- Connect it to the data source node.
- Configure the node to display the dataset as an interactive table.
- Create a dashboard by adding interactive elements (e.g., filter widgets, bar charts, or line charts).
- Link these elements to the interactive table for dynamic data exploration.
- Execute the workflow to open and interact with the dashboard.

Model Deployment and Integration:

- Add a machine learning model node (e.g., "Decision Tree Predictor" or any other model) to the workflow.
- Train the model using a portion of the dataset.
- Add a "Model Writer" node to save the trained model to a file.
- Use a "Model Reader" node to load the trained model from the saved file.
- Configure the "Model Reader" node to apply the model to the remaining dataset.
- Execute the workflow to deploy and integrate the model for predictions.

Automation and Reporting in KNIME:

- Add a "Reporting" node to the workflow.
- Configure the reporting node to generate a report summarizing the analysis.
- Include visualizations, summary statistics, and insights in the report.
- Save the report to a specified location.
- Use a "Send Email" node to automate the delivery of the report to stakeholders.
- Execute the workflow to generate and send the report.

Validation and Reflection:

- Review the generated scatter plot, interactive dashboard, and deployed model predictions.
- Examine the automated report and verify its content.
- Reflect on the exercise and how these visualization, dashboard, deployment, and automation techniques can be applied to real-world data analysis projects.
- Consider other scenarios where KNIME's visualization, dashboarding, deployment, and automation capabilities can be beneficial.

This exercise provides a hands-on experience with KNIME, covering data visualization, dashboard creation, model deployment, and automation, allowing you to leverage KNIME effectively for data-driven decision-making and reporting.

SUMMARY

In conclusion, the fusion of Visualization and Deployment capabilities within KNIME empowers data professionals to bridge the gap between data insights and actionable outcomes. Effective data visualization in KNIME transforms complex datasets into intuitive visual representations, enabling stakeholders to grasp trends and patterns at a glance. The creation of interactive dashboards further enhances data exploration by allowing users to interact with data dynamically, fostering a deeper understanding of information. On the deployment front, KNIME facilitates the seamless integration of machine learning models into real-world applications, translating predictive insights into tangible business value. Moreover, the automation and reporting features in KNIME streamline data analysis workflows, ensuring that critical insights are delivered to decision-makers efficiently and consistently. This combination of visualization and deployment capabilities positions KNIME as a powerful tool in the arsenal of data professionals, driving informed decision-making, innovation, and competitive advantage in an increasingly data-centric world.

REFERENCE

1 *https://www.knime.com/*
2 *https://www.knime.com/getting-started-guide*
3 *https://www.knime.com/knimepress*
4 *KNIME Beginner's Luck Book by Rosario_Silipo*
5 *KNIME Advanced Luck A Guide to KNIME Analytics Platform for Advanced Users by Rosario Silipo, Jeanette Prinz*

MULTIPLE CHOICE QUESTIONS (MCQ) FOR PRACTICE

1 What is the primary purpose of data visualization in KNIME?
   a) To generate random data
   b) To create 3D animations
   c) To convey insights through visual representations
   d) To send emails

2 Which of the following is NOT a common type of data visualization in KNIME?
   a) Bar chart
   b) Scatter plot
   c) Musical composition
   d) Line chart

3 In KNIME, what do interactive dashboards allow users to do?
   a) Watch videos
   b) Explore data dynamically and interact with visual elements
   c) Create 3D models
   d) Change node colors

4 What is the primary goal of model deployment and integration in KNIME?
   a) To create data visualizations
   b) To send emails to stakeholders
   c) To translate predictive models into real-world applications
   d) To perform data transformations

5 How does KNIME facilitate the deployment of machine learning models?
   a) By creating animated videos
   b) By generating random data
   c) By providing integration options for deploying models in production environments
   d) By changing font size

6 What does automation in KNIME refer to?
   a) The process of creating interactive dashboards
   b) The process of generating random data
   c) The automatic execution of workflows and tasks
   d) The creation of 3D animations

7 Which of the following is a benefit of using interactive dashboards in data analysis?
   a) Slower data exploration
   b) Deeper insights through user interaction
   c) Increased complexity
   d) Deleting data

8 In KNIME, what is the purpose of reporting features?
   a) To change node colors
   b) To create interactive dashboards
   c) To streamline data analysis workflows and generate reports
   d) To send emails to colleagues

9 Which visualization type is often used to display trends over time in KNIME?
   a) Pie chart
   b) Bar chart
   c) Line chart
   d) Scatter plot

10 What is the primary advantage of interactive dashboards in KNIME?
   a) They make data exploration static and non-interactive.
   b) They allow users to explore data dynamically and gain deeper insights.
   c) They generate random data.
   d) They create 3D animations.

11 How does model deployment support decision-making in real-world applications?
   a) It hinders the deployment of models.
   b) It allows models to remain in a testing phase.
   c) It translates predictive insights into actionable outcomes.
   d) It generates random data for decision-making.

12 Which of the following is NOT a common visualization element in KNIME?
   a) Data tables
   b) Interactive sliders
   c) Augmented reality simulations
   d) Bar charts

13 What role does automation play in reporting in KNIME?
   a) It creates animated videos.
   b) It sends emails to stakeholders.
   c) It streamlines data analysis workflows by automating repetitive tasks.
   d) It generates random data for reports.

14 In KNIME, how can you ensure that reports are generated consistently and on schedule?
   a) By manually generating reports when needed
   b) By changing node colors
   c) By automating report generation tasks
   d) By sending emails to the reporting team

15 What is the primary purpose of integrating machine learning models into real-world applications in KNIME?
   a) To create interactive dashboards
   b) To visualize data patterns
   c) To translate predictive insights into practical decision support
   d) To generate random data for applications

Answers

1 *c) To convey insights through visual representations*
2 *c) Musical composition*
3 *b) Explore data dynamically and interact with visual elements*
4 *c) To translate predictive models into real-world applications*
5 *c) By providing integration options for deploying models in production environments*
6 *c) The automatic execution of workflows and tasks*
7 *b) Deeper insights through user interaction*
8 *c) To streamline data analysis workflows and generate reports*
9 *c) Line chart*
10 *b) They allow users to explore data dynamically and gain deeper insights.*
11 *c) It translates predictive insights into actionable outcomes.*
12 *c) Augmented reality simulations*
13 *c) It streamlines data analysis workflows by automating repetitive tasks.*
14 *c) By automating report generation tasks*
15 *c) To translate predictive insights into practical decision support*

PRACTICAL USE CASES

I **BUDGET MONITORING REPORT**

This workflow accesses an accounting of a list of projects, on one branch it calculates the money in budget for each year for each one of the project and on the other branch it calculates the money actually spent for each project each year.
It then reports the money allocated, the money spent and the money remaining for each project each year.

II **DECISION TREE**

The dataset we use in this exercise describes the sale of individual residential properties in Ames, Iowa from 2006 to 2010. One of the columns is the overall condition ranking, with values between 1 and 10.
The goal of this exercise is to train a binary classification model, which can predict whether the overall condition is high or low. To do so, the workflow below reads the data set and creates the class column based on overall condition ranking, which is called rank and has the values low if the overall condition is smaller or equal to 5, otherwise high.

III **VISUALIZATION OF SALES DATA**

- Filter data to columns that contain relevant information
- Filter data to rows that you want to include in your analysis
- Visualize the data using a Stacked Area Chart and a Pie/Donut Chart

IV DATA BLENDING AND ETL

Blend together many different data sources through an effective ETL (Extract, Transform and Loading).
This workflow illustrates how to blend different data sources within KNIME Analytics Platform. A Java Script node is used to visualize average age vs. product in the resulting blended table.

V CREDIT SCORING

Credit scoring is a technique used to determine whether or not to extend credit (and if so, how much) to a borrower. This workflow illustrates how to create and choose a credit scoring model based on both historical data and on the application of different machine learning algorithms.

Task Create a credit scoring model based on historical data. Select the best machine learning algorithm to be applied. Use cross-validation to evaluate model performance.

PYTHON

TASK STATEMENT

1. Write simple Python scripts to perform common tasks, such as arithmetic operations, string manipulation, and basic input / output operations.
2. Utilize built-in data structures in Python, including lists, tuples, dictionaries, to store and manipulate data.
3. Evaluate Loops and Conditional Statements.
4. Develop problem-solving skills by applying Python programming concepts to solve simple programming challenges and exercise.

KNOWLEDGE STATEMENT

1. Understanding the knowledge of fundamental programming concepts such as variables, data types, operators, and control structures (e.g., loops, conditionals).
2. Learning the Fundamental concepts in python like Variable, Expressions, Strings
3. Recognize the importance of functions and be able to define and call functions in Python for code reusability and modularization in the problems.

KNIME

TASK STATEMENT

1. Install KNIME Analytics Platform and necessary extensions.
2. Create a workspace and import a workflow for analysis.
3. Import data, clean it (delete rows/columns), and perform basic manipulations (e.g., IF function).
4. Establish a connection to a Big Data source and configure nodes for data extraction.
5. Add a machine learning node to create a predictive analysis model.
6. Create interactive dashboards for visualization and generate reports using the reporting node.
7. Send the report as an email to stakeholders for review and feedback.

KNOWLEDGE STATEMENT

1. Installation and System Requirements for KNIME
2. Searching and Using Nodes in Node Repository
3. Data Wrangling Techniques in KNIME
4. Applying Machine Learning Techniques
5. Visualization in KNIME/Power BI

य एष सुप्तेषु जागर्ति

ISBN No- 978-81-19472-79-6

Board of Studies (Academic)
The Institute of Chartered Accountants of India
(Setup by an Act of Parliament)

ICAI Bhawan, A-29, Sector-62, Noida 201 309
Phone: 0120-3045964