Use this guide as a reference if you need additional context or clarification as you watch the next video. If you'd like to follow along with the video, choose a spreadsheet tool and open a blank sheet.

Example 1: Get started

Enter basic data:

1. Begin with a new spreadsheet.
2. Select cell **A2**.
3. Enter your **first name**.
4. Select cell **B2**.
5. Enter your **last name**.

Adjust the size of rows and columns:
To make the text fit in the rows and columns, adjust their sizes. Use either of the following methods:

1. If your name is longer than the width of the column, **select and drag** the right edge of the corresponding column until it fits.
2. To **wrap text**, select the cells, columns, or rows with text that you want to reformat.
3. Select the **Format** menu.
4. Under **Wrapping**, select **Wrap**.

Example 2: Add labels

Add labels, or attributes, to help you keep track of the data:

1. Select cell **A1**.
2. Enter **First Name**.
3. Select cell **B1**.
4. Enter **Last Name**.
5. Select cells **A1** and **B1**. To do this, select a single cell and drag your cursor over to the other cell to include it in the selection.
6. From the toolbar, select the **bold icon**.

Example 3: Add more attributes and data

Add more attributes and data to your spreadsheet:

1. Select cell **C1** and enter **Siblings**.
2. Select cell **D1** and enter **Favorite Color**.
3. Select cell **E1** and enter **Favorite Dessert**.
4. Select **all three cells** and make them bold by selecting the **bold icon** from the toolbar.
5. Adjust the columns to fit the new text.
6. Enter the corresponding data in cells **C2**, **D2**, and **E2** (your number of siblings, favorite color, and favorite dessert).

7. Add data about two more people in **rows 3 and 4**. These can be people you know or people you've just made up.

Example 4: Organize your data

One way to organize your data is by sorting it.

1. Select all columns that contain data. There are a few ways to select multiple cells:

    a. To select nonadjacent cells and/or cell ranges, hold the **Command** (Mac) or **Ctrl** (PC) **key** and select the cells.
    b. To select a range of cells, hold the **Shift** key and either drag your cursor over which cells you want to include or use the arrow keys to select a range.

    c. Select a single cell and drag your cursor over the cells you want to include in your selection.
2. Select the **Data** menu.
3. Select **Sort range**, then select **Advanced range sorting options**.
4. In the **Advanced range sorting options** window, select the checkbox for **Data has header row**. Make sure that **A to Z** is selected.
5. Select the **Sort by** drop-down menu, then select **Siblings**.
6. Select **Sort**. This will organize the spreadsheet by the number of siblings, from lowest to highest.

Example 5: Use a formula

Spreadsheets enable data professionals to analyze data. In this example, the instructor uses a formula to calculate a sum.

1. Select the next empty cell in the **Siblings** column (**C5**).
2. Enter the formula `=C2+C3+C4`.
3. Press **Enter** on your keyboard to complete the formula.
4. The formula calculates the total number of siblings.

Mark as completed
Like
Dislike
Report an issue

Activity Overview

At this point in the course, you've explored how spreadsheets can be used to support data-driven business decisions. In this activity, you will use a spreadsheet to make a simple chart. By the time you're done, you'll understand how to create a simple graphical representation of information. Data analysts use this skill to make reports, presentations, infographics, and more engaging and accessible.

Scenario

Review the following scenario. Then complete the step-by-step instructions.

The scenario: Analyze patterns in monthly sales

To help determine optimal inventory and staffing levels, your company has asked you to analyze total sales trends for the last three years. Because your firm relies on tourists for the bulk of its sales, leaders know that demand for inventory and staffing requirements vary based on the tourist season. They have asked you to identify peak months to help with forecasting requirements for next year. To do this, you'll create a chart.

Step-By-Step Instructions

Working with spreadsheets

Follow the instructions to complete each step of the activity. Then answer the questions at the end of the activity before going to the next course item.

Step 1: Access the template

The monthly sales spreadsheet

To use the spreadsheet for this course item, select the link below, then select the "Use Template" button to open your own version of the spreadsheet.

Link to template: Monthly sales

OR

If you don't have a Google account, download the template directly by selecting the attachment below.

Monthly sales

XLSX File

Save the spreadsheet using a preferred file naming convention and store it in a folder to help you stay organized.

Step 2: Open the spreadsheet

If you haven't done so already, open the Monthly sales spreadsheet.

Step 3: Review the data

This spreadsheet contains total sales data for each month beginning in January 2021 and ending in December 2023. A portion of the spreadsheet is shown below.

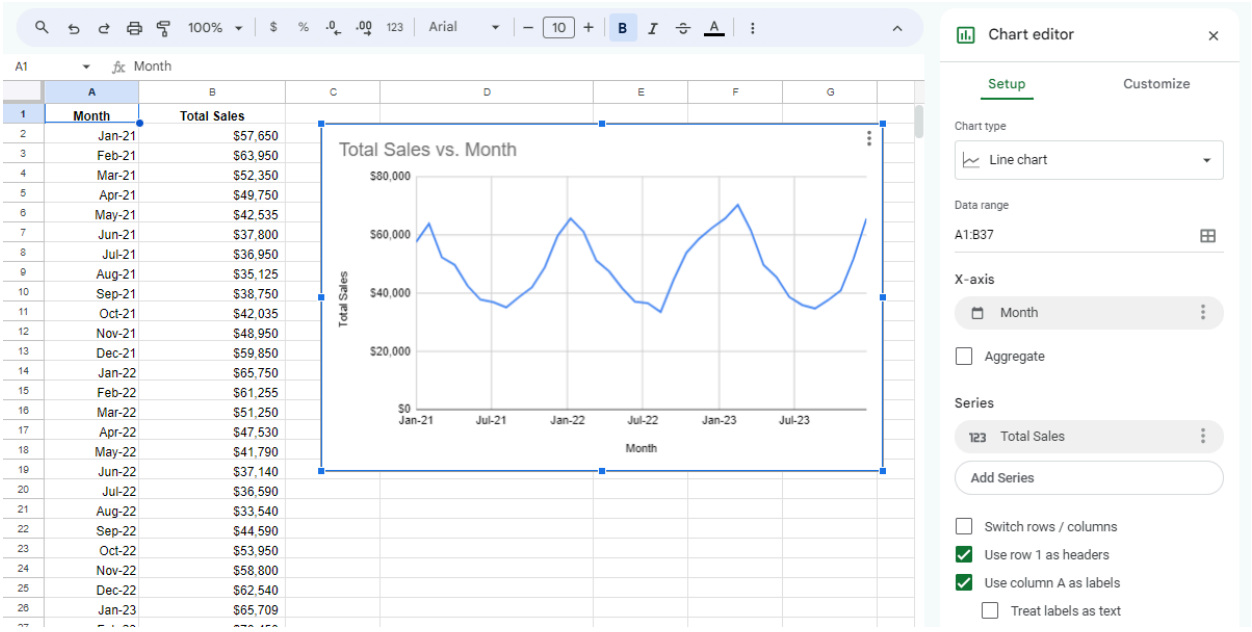
A1 Month

	A	B
1	Month	Total Sales
2	Jan-21	$57,650
3	Feb-21	$63,950
4	Mar-21	$52,350
5	Apr-21	$49,750
6	May-21	$42,535
7	Jun-21	$37,800
8	Jul-21	$36,950
9	Aug-21	$35,125
10	Sep-21	$38,750
11	Oct-21	$42,035
12	Nov-21	$48,950
13	Dec-21	$59,850
14	Jan-22	$65,750

Step 4: Create a chart

To make it easier to understand monthly sales trends, use these steps to create a chart:

1. Select cell A1.
2. Choose Insert > Chart. Google Sheets inserts a chart to the right of the data. By default, Google Sheets creates a line chart.
3. Notice that Google Sheets generated a title automatically for the chart, “Total Sales vs. Month,” and added labels of “Total Sales,” for the vertical axis, and “Month” for the horizontal axis. It also automatically opens the Chart editor panel on the right side of the window, with the Setup tab opened.



The trend appears to be flat overall, varying between about $40,000 and $60,000 with consistent peaks near January and consistent lows near August.
This is a simple example of how you can use a chart to visualize data in a spreadsheet. You'll learn more about creating, modifying, and interpreting data visualizations in an upcoming lesson.

Step 5: Adjust the chart

Use the Chart editor panel to make changes to the chart. In the Setup tab, you can change the chart type, data range, axis labels, and more. For example, the chart type drop-down list changes the type of chart from line chart to a different style, such as a bar chart. In the Customize tab, you can change many other options, such as the chart's style.
If the Chart editor panel isn't open, double-click on the chart to open it. Then, change the various components of the chart by selecting them within the chart or by navigating to them from the Chart editor panel.
In this example, change the chart title to "Monthly sales." Follow these steps:

1. If necessary, open the Chart editor panel by double-clicking on the chart.
2. In the Chart editor panel, select the Customize tab.
3. Expand the Chart & axis titles section.
4. Verify that the Chart title is selected from the dropdown, then change the title to Monthly sales. Your chart will nowdisplay the title "Monthly sales"
5. If you're using Microsoft Excel, double-click on the title in the chart to edit directly.

Get creative! Don't be afraid to play around with the options. You can always save or download another copy of the template if you want to start fresh!

Pro Tip: Save the activity spreadsheet template

Be sure to save a copy of the spreadsheet template you used to complete this activity. You can use it for further practice or to help you work through your thought processes for similar tasks in a future data analyst role.

More spreadsheet resources

In the spirit of lifelong learning, it is good to have resources to turn to when you want to know more about using spreadsheets. Two of the most well known and used spreadsheet platforms are Google Sheets and Microsoft Excel. Both provide free online training resources that you can access anytime you need them. Bookmark these links if you want to access them later.

Google Sheets Training and Help

Learn even more ways to move, store, and analyze your data with the Google Sheets Training and Help page, located in the Google Workspace Learning Center. This hub offers an expanded list of tips, from beginner to advanced, along with cheat sheets, templates, guides, and tutorials.

Google Sheets Quick Tips

Want to learn more about Google Sheets? This online help article features a short list of the most important functions you will use, including rows, columns, cells, and functions.

Microsoft Excel for Windows Training

Get to know Excel spreadsheets a little better by visiting this free online training center. Offering everything from a quick-start guide and introduction to tutorials and templates, you will find everything you need to know, all in one place.

SQL guide: Getting started

Just as humans use different languages to communicate with others, so do computers. **Structured Query Language** (or **SQL**, often pronounced “sequel”) enables data analysts to talk to their databases. SQL is one of the most useful data analyst tools, especially when working with large datasets in tables. It can help you investigate huge databases, track down text (referred to as strings) and numbers, and filter for the exact kind of data you need—much faster than a spreadsheet can.

If you haven’t used SQL before, this reading will help you learn the basics so you can appreciate how useful SQL is and how useful SQL queries are in particular. You will be writing SQL queries in no time at all.

What is a query?

A **query** is a request for data or information from a database. When you query databases, you use SQL to communicate your question or request. You and the database can always exchange information as long as you speak the same language.

Every programming language, including SQL, follows a unique set of guidelines known as **syntax**. Syntax is the predetermined structure of a language that includes all required words, symbols, and punctuation, as well as their proper placement. As soon as you enter your search criteria using the correct syntax, the query starts working to pull the data you’ve requested from the target database.

The syntax of every SQL query is the same:

- Use `SELECT` to choose the columns you want to return.
- Use `FROM` to choose the tables where the columns you want are located.
- Use `WHERE` to filter for certain information.

A SQL query is like filling in a template. You will find that if you are writing a SQL query from scratch, it is helpful to start a query by writing the `SELECT`, `FROM`, and `WHERE` keywords in the following format:

```
SELECT
FROM
WHERE
```

Next, enter the table name after the `FROM`; the table columns you want after the `SELECT`; and, finally, the conditions you want to place on your query after the `WHERE`. Make sure to add a new line and indent when adding these, as shown below:

`SELECT`	Specifies the columns from which to retrieve data
`FROM`	Specifies the table from which to retrieve data
`WHERE`	Specifies criteria that the data must meet

Following this method each time makes it easier to write SQL queries. It can also help you make fewer syntax errors.

Example of a query

Here is how a simple query would appear in BigQuery, a data warehouse on the Google Cloud Platform.

```
2

3

SELECT first_name

FROM customer_data.customer_name

WHERE first_name = 'Tony'
```

The above query uses three commands to locate customers with the **`first_name`**, **`'Tony'`**:

1. **`SELECT`** the column named **`first_name`**

`FROM` a table named **`customer_name`** (in a dataset named **`customer_data`**)

2. (The dataset name is always followed by a dot, and then the table name.)
3. But only return the data **`WHERE`** the **`first_name`** is **`'Tony'`**

The results from the query might be similar to the following:

first_name
Tony
Tony
Tony

As you can conclude, this query had the correct syntax, but wasn't very useful after the data was returned.

Multiple columns in a query

Of course, as a data professional, you will need to work with more data beyond customers named Tony. Multiple columns that are chosen by the same **`SELECT`** command can be indented and grouped together.
If you are requesting multiple data fields from a table, you need to include these columns in your **`SELECT`** command. Each column is separated by a comma as shown below:

```
4

5

6

7
```

```
    ColumnC
FROM
    Table where the data lives
WHERE
    Certain condition is met
```

Here is an example of how it would appear in BigQuery:

```
    first_name = 'Tony'
```

The above query uses three commands to locate customers with the **`first_name`**, **`'Tony'`**.

1. **`SELECT`** the columns named **`customer_id`**, **`first_name`**, and **`last_name`**

`FROM` a table named **`customer_name`** (in a dataset named **`customer_data`**)

2. (The dataset name is always followed by a dot, and then the table name.)
3. But only return the data **`WHERE`** the **`first_name`** is **`'Tony'`**

The only difference between this query and the previous one is that more data columns are selected. The previous query selected **`first_name`** only while this query selects **`customer_id`** and **`last_name`** in addition to **`first_name`**. In general, it is a more efficient use of resources to select only the columns that you need. For example, it makes sense to select more columns if you will actually use the additional fields in your **`WHERE`** clause. If you have multiple conditions in your **`WHERE`** clause, they may be written like this:

```
SELECT

ColumnA,

ColumnB,

ColumnC

FROM

    Table where the data lives

WHERE

    Condition 1

    AND Condition 2

    AND Condition 3
```

Notice that unlike the **SELECT** command that uses a comma to separate fields / variables / parameters, the **WHERE** command uses the **AND** statement to connect conditions. As you become a more advanced writer of queries, you will make use of other connectors / operators such as **OR** and **NOT**.
Here is a BigQuery example with multiple fields used in a **WHERE** clause:

```
6
7
8
9
10
```

```
SELECT

    customer_id,

    first_name,

    last_name

FROM

    customer_data.customer_name

WHERE

    customer_id > 0

    AND first_name = 'Tony'

    AND last_name = 'Magnolia'
```

The above query uses three commands to locate customers with a valid (greater than 0), **`customer_id`** whose **`first_name`** is **`'Tony'`** and **`last_name`** is **`'Magnolia'`**.

1. **`SELECT`** the columns named **`customer_id`**, **`first_name`**, and **`last_name`**

`FROM` a table named **`customer_name`** (in a dataset named **`customer_data`**)

2. (The dataset name is always followed by a dot, and then the table name.)
3. But only return the data **`WHERE`** **`customer_id`** is greater than **`0`**, **`first_name`** is **`Tony`**, and **`last_name`** is **`Magnolia`**.

Note that one of the conditions is a logical condition that checks to see if **`customer_id`** is greater than zero.
If only one customer is named Tony Magnolia, the results from the query could be:

customer_id	first_name	last_name
1967	Tony	Magnolia

If more than one customer has the same name, the results from the query could be:

customer_id	first_name	last_name
1967	Tony	Magnolia
7689	Tony	Magnolia

Key takeaways

The `SELECT`, `FROM`, and `WHERE` clauses are the essential building blocks of SQL queries. Queries with multiple fields will become simpler after you practice writing your own SQL queries later in the program.

Endless SQL possibilities

You have learned that a SQL query uses **SELECT**, **FROM**, and **WHERE** to specify the data to be returned from the query. This reading provides more detailed information about formatting queries, using **WHERE** conditions, selecting all columns in a table, adding comments, and using aliases. All of these make it easier for you to understand (and write) queries to put SQL in action. The last section of this reading provides an example of what a data analyst would do to pull employee data for a project.

Capitalization, indentation, and semicolons

You can write your SQL queries in all lowercase and don't have to worry about extra spaces between words. However, using capitalization and indentation can help you read the information more easily. Keep your queries neat, and they will be easier to review or troubleshoot if you need to check them later on.

```
SELECT field1
FROM table
WHERE field1 = condition;
```

Notice that the SQL statement shown above has a semicolon at the end. The semicolon is a statement terminator and is part of the American National Standards Institute (ANSI) SQL-92 standard, which is a recommended common syntax for adoption by all SQL databases. However, not all SQL databases have adopted or enforce the semicolon, so it's possible you may come across some SQL statements that aren't terminated with a semicolon. If a statement works without a semicolon, it's fine.

WHERE conditions

In the query shown above, the **SELECT** clause identifies the column you want to pull data from by name, **field1**, and the **FROM** clause identifies the **table** where the column is located by name, table. Finally, the **WHERE** clause narrows your query so that the database returns only the data with an exact value match or the data that matches a certain condition that you want to satisfy.
For example, if you are looking for a specific customer with the last name Chavez, the **WHERE** clause would be:
WHERE field1 = 'Chavez'
However, if you are looking for all customers with a last name that begins with the letters "Ch," the **WHERE** clause would be:

`WHERE field1 LIKE 'Ch%'`

You can conclude that the **`LIKE`** clause is very powerful because it allows you to tell the database to look for a certain pattern! The percent sign `%` is used as a wildcard to match one or more characters. In the example above, both **Chavez** and **Chen** would be returned. Note that in some databases an asterisk **`*`** is used as the wildcard instead of a percent sign `%`.

SELECT all columns

Can you use **`SELECT *`** ?

In the example, if you replace **`SELECT field1`** with **`SELECT *`** , you would be selecting all of the columns in the table instead of the field1 column only. From a syntax point of view, it is a correct SQL statement, but you should use the asterisk **`*`** sparingly and with caution. Depending on how many columns a table has, you could be selecting a tremendous amount of data. Selecting too much data can cause a query to run slowly.

Comments

Some tables aren't designed with descriptive enough naming conventions. In the example, **`field1`** was the column for a customer's last name, but you wouldn't know it by the name. A better name would have been something such as **`last_name`**. In these cases, you can place comments alongside your SQL to help you remember what the name represents. Comments are text placed between certain characters, **`/*`** and **`*/`**, or after two dashes **`--`**) as shown below.

```
1
2
3
4
5
6
```

```
SELECT
    field1 /* this is the last name column */
FROM
    table -- this is the customer data table
WHERE
```

```
    field1 LIKE 'Ch%';
```

Comments can also be added outside of a statement as well as within a statement. You can use this flexibility to provide an overall description of what you are going to do, step-by-step notes about how you achieve it, and why you set different parameters/conditions.

```
7
6
3
4
5
1
2
```

```
FROM  Publishers
```

```
        rowkey,  -- key used to join with account_id
Info.date,  -- date is in string format YYYY-MM-DD HH:MM:SS
Info.code  -- e.g., 'pub-###'
-- This is an important query used later to join with the accounts table
SELECT
```

The more comfortable you get with SQL, the easier it will be to read and understand queries at a glance. Still, it never hurts to have comments in a query to remind yourself of what you're trying to do. This also makes it easier for others to understand your query if your query is shared. As your queries become more and more complex, this practice will save you a lot of time and energy to understand complex queries you wrote months or years ago.

Example of a query with comments

Here is an example of how comments could be written in BigQuery:

```
1
```

```
2
3
4
5
6
7
```

```
-- Pull basic information from the customer table

SELECT

    customer_id, --main ID used to join with customer_addresss

    first_name, --customer's first name from loyalty program

    last_name --customer's last name

FROM

    customer_data.customer_name
```

In the above example, a comment has been added before the SQL statement to explain what the query does. Additionally, a comment has been added next to each of the column names to describe the column and its use. Two dashes `--` are generally supported. So it is best to use `--` and be consistent with it. You can use `#` in place of `--` in the above query, but `#` is not recognized in all SQL versions; for example, MySQL doesn't recognize `#`. You can also place comments between `/*` and `*/` if the database you are using supports it.
As you develop your skills professionally, depending on the SQL database you use, you can pick the appropriate comment delimiting symbols you prefer and stick with those as a consistent style. As your queries become more and more complex, the practice of adding helpful comments will save you a lot of time and energy to understand queries that you may have written months or years prior.

Aliases

You can also make it easier on yourself by assigning a new name or **alias** to the column or table names to make them easier to work with (and avoid the need for comments). This is done with a SQL AS clause. In the example below, aliases are used for both a table name and a column. Within the database, the table is called **`actual_table_name`** and the column in that table is called

actual_column_name. They are aliased as **my_table_alias** and **my_column_alias**, respectively. These aliases are good for the duration of the query only. An alias doesn't change the actual name of a column or table in the database.

Example of a query with aliases

```
SELECT
    my_table_alias.actual_column_name AS my_column_alias
FROM
    actual_table_name AS my_table_alias
```

Putting SQL to work as a data analyst

Imagine you are a data analyst for a small business and your manager asks you for some employee data. You decide to write a query with SQL to get what you need from the database.
You want to pull all the columns: **empID, firstName, lastName, jobCode,** and **salary.** Because you know the database isn't that big, instead of entering each column name in the **SELECT** clause, you use **SELECT *.** This will select all the columns from the Employee table in the **FROM** clause.

```
SELECT
    *
FROM
```

```
    Employee
```

Now, you can get more specific about the data you want from the **Employee** table. If you want all the data about employees working in the `'SFI'` job code, you can use a **`WHERE`** clause to filter out the data based on this additional requirement.
Here, you use:

```
SELECT
    *
FROM
    Employee
WHERE
    jobCode = 'SFI'
```

A portion of the resulting data returned from the SQL query might look like this:

empID	firstName	lastName	jobCode	salary
0002	Homer	Simpson	SFI	15000
0003	Marge	Simpson	SFI	30000
0034	Bart	Simpson	SFI	25000
0067	Lisa	Simpson	SFI	38000
0088	Ned	Flanders	SFI	42000

0076	Barney	Gumble	SFI	32000

Suppose you notice a large salary range for the `'SFI'` job code. You might like to flag all employees in all departments with lower salaries for your manager. Because interns are also included in the table and they have salaries less than $30,000, you want to make sure your results give you only the full time employees with salaries that are $30,000 or less. In other words, you want to exclude interns with the `'INT'` job code who also earn less than $30,000. The **`AND`** clause enables you to test for both conditions.
You create a SQL query similar to below, where <> means "does not equal":

```
SELECT
    *
FROM
    Employee
WHERE
    jobCode <> 'INT'
      AND salary <= 30000;
```

The resulting data from the SQL query might look like the following (interns with the job code **INT** aren't returned):

empID	firstName	lastName	jobCode	salary
0002	Homer	Simpson	SFI	15000

0003	Marge	Simpson	SFI	30000
0034	Bart	Simpson	SFI	25000
0108	Edna	Krabappel	TUL	18000
0099	Moe	Szyslak	ANA	28000

With quick access to this kind of data using SQL, you can provide your manager with tons of different insights about employee data, including whether employee salaries across the business are equitable. Fortunately, the query shows only an additional two employees might need a salary adjustment and you share the results with your manager.
Pulling the data, analyzing it, and implementing a solution might ultimately help improve employee satisfaction and loyalty. That makes SQL a pretty powerful tool.

Resources to learn more

Nonsubscribers may access these resources for free, but if a site limits the number of free articles per month and you already reached your limit, bookmark the resource and come back to it later.

- W3Schools SQL Tutorial: If you would like to explore a detailed tutorial of SQL, this is the perfect place to start. This tutorial includes interactive examples you can edit, test, and recreate. Use it as a reference or complete the whole tutorial to practice using SQL. Click the green **Start learning SQL now** button or the **Next** button to begin the tutorial.
- SQL Cheat Sheet: For more advanced learners, go through this handy 3-page resource to gain an overview of additional SQL functions and formulas. By the time you are finished looking through the cheat sheet, you will know a lot more about the various SQL techniques and will be prepared to use it for business analysis and other tasks.

Key takeaways

SQL queries use `SELECT`, `FROM`, and `WHERE` to specify the data to be returned from the query. Capitalization, indentation, and semicolons are useful for making your SQL queries easier to read. In addition, comments can be added to explain queries to others. As you progress through this course, you will continue discovering many ways in which SQL can be a very powerful tool for retrieving, analyzing, and interpreting data.

Plan a data visualization

Earlier, you learned that **data visualization** is the graphical representation of information. As a data analyst, you will want to create visualizations that make your data easy to understand and interesting to look at. Because of the importance of data visualization, most data analytics tools (such as spreadsheets and databases) have a built-in visualization component while others (such as Tableau) specialize in visualization as their primary value-add. In this reading, you will explore the steps involved in the data visualization process and a few of the most common data visualization tools available.

Steps to plan a data visualization

Let's go through an example of a real-life situation where a data analyst might need to create a data visualization to share with stakeholders. Imagine you're a data analyst for a clothing distributor. The company helps small clothing stores manage their inventory, and sales are booming. One day, you learn that your company is getting ready to make a major update to its website. To guide decisions for the website update, you're asked to analyze data from the existing website and sales records. Let's go through the steps you might follow.

Step 1: Explore the data for patterns

First, you ask your manager or the data owner for access to the current sales records and website analytics reports. This includes information about how customers behave on the company's existing website, basic information about who visited, who bought from the company, and how much they bought.

While reviewing the data you notice a pattern among those who visit the company's website most frequently: geography and larger amounts spent on purchases. With further analysis, this information might explain why sales are so strong right now in the northeast—and help your company find ways to make them even stronger through the new website.

Step 2: Plan your visuals

Next it is time to refine the data and present the results of your analysis. Right now, you have a lot of data spread across several different tables, which isn't an ideal way to share your results with management and the marketing team. You will want to create a data visualization that explains your

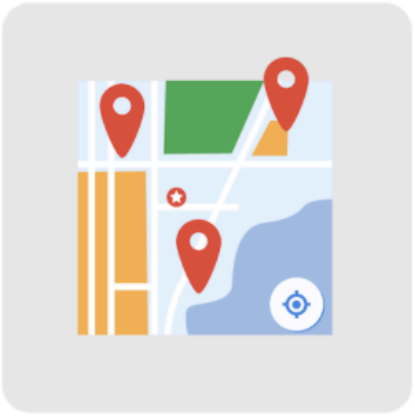
findings quickly and effectively to your target audience. Since you know your audience is sales oriented, you already know that the data visualization you use should:

- Show sales numbers over time
- Connect sales to location
- Show the relationship between sales and website use
- Show which customers fuel growth

Step 3: Create your visuals

Now that you have decided what kind of information and insights you want to display, it is time to start creating the actual visualizations. Keep in mind that creating the right visualization for a presentation or to share with stakeholders is a process. It involves trying different visualization formats and making adjustments until you get what you are looking for. In this case, a mix of different visuals will best communicate your findings and turn your analysis into the most compelling story for stakeholders. So, you can use the built-in chart capabilities in your spreadsheets to organize the data and create your visuals.

Line charts can track sales over time



Maps can connect sales to locations

Donut charts can show customer segments

Bar charts can compare total visitors and visitors that make a purchase

1) line charts can track sales over time

2) maps can connect sales to locations
3) donut charts can show customer segments
4) bar charts can compare total visitors that make a purchase

Build your data visualization toolkit

There are many different tools you can use for data visualization.

- You can use the visualizations tools in your spreadsheet to create simple visualizations such as line and bar charts.
- You can use more advanced tools such as Tableau that allow you to integrate data into dashboard-style visualizations.
- If you're working with the programming language R you can use the visualization tools in RStudio.

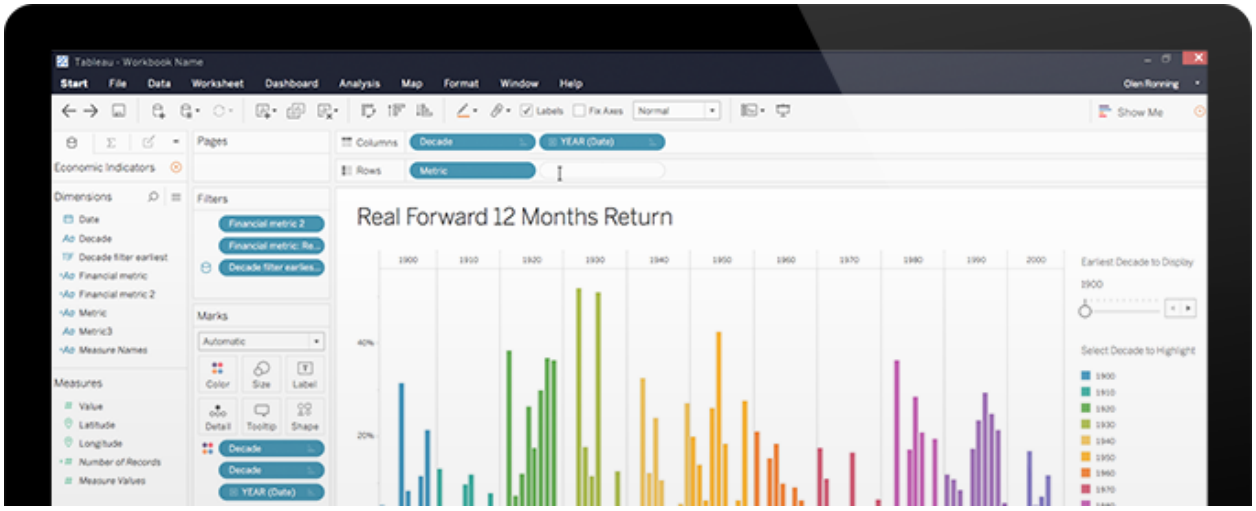
Your choice of visualization will be driven by a variety of drivers including the size of your data, the process you used for analyzing your data (spreadsheet, or databases/queries, or programming languages). For now, just consider the basics.

Spreadsheets (Microsoft Excel or Google Sheets)

In our example, the built-in charts and graphs in spreadsheets made the process of creating visuals quick and easy. Spreadsheets are great for creating simple visualizations like bar graphs and pie charts, and even provide some advanced visualizations like maps, and waterfall and funnel diagrams (shown in the following figures).
But sometimes you need a more powerful tool to truly bring your data to life. Tableau and RStudio are two examples of widely used platforms that can help you plan, create, and present effective and compelling data visualizations.

Visualization software (Tableau)

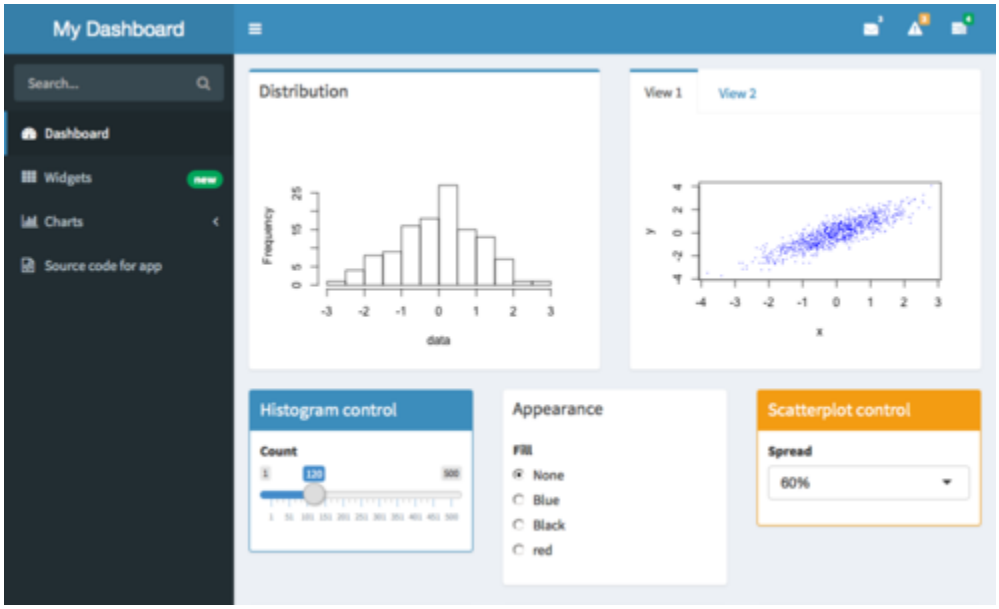
Tableau is a popular data visualization tool that lets you pull data from nearly any system and turn it into compelling visuals or actionable insights. The platform offers built-in visual best practices, which makes analyzing and sharing data fast, easy, and (most importantly) useful. Tableau works well with a wide variety of data and includes an interactive dashboard that lets you and your stakeholders click to explore the data interactively.



You can start exploring Tableau from the How-to Video resources. Tableau Public is free, easy to use, and full of helpful information. The Resources page is a one-stop-shop for how-to videos, examples, and datasets for you to practice with. To explore what other data analysts are sharing on Tableau, visit the Viz of the Day page where you will find beautiful visuals ranging from an overview of the Lighthouses of Greece to Who's Talking in Popular Films.

Programming language (R with RStudio)

A lot of data analysts work with a programming language called R. Most people who work with R end up also using RStudio, an integrated developer environment (IDE), for their data visualization needs. As with Tableau, you can create dashboard-style data visualizations using RStudio.



Check out their website to learn more about RStudio.
You could easily spend days exploring all the resources provided at RStudio.com, but the RStudio Cheatsheets and the RStudio Visualize Data Primer are great places to start. When you have more time, check out the webinars and videos which offer advice and helpful perspectives for both beginners and advanced users.

Key takeaways

The best data analysts use lots of different tools and methods to visualize and share their data. As you continue learning more about data visualization throughout this course, be sure to stay curious, research different options, and continuously test new programs and platforms to help you make the most of your data.

1. In this spreadsheet, what type of tree is in cell C2? 1 point

(n/a)	A	B	C	D	E
1	Ginkgo	Weeping willow	Sycamore	Bay laurel	Pistachio
2	Papaya	Maple	White oak	European ash	Pecan
3	Cedar	Burflower	Spruce	Redwood	Beech
4	Birch	Cottonwood	Clove	Cacao	Bristlecone
5	Rubber tree	Brazil nut	Walnut	Pine	Palm

- ○ Sycamore
- ◉ White oak
- ○ Palm
- ○ Maple

2. In the following query, what will be retrieved from the database? 1 point

```
SELECT *
FROM Riverside Elementary
WHERE grade = 'kindergarten'
```

- ○ All kindergarteners in all elementary schools
- ◉ All kindergarteners at Riverside Elementary
- ○ All grades at Riverside Elementary
- ○ All students and grades at Riverside Elementary

3. Which of the following statements accurately describe spreadsheet attributes and observations? Select all that apply. 1 point

- ☐ An attribute includes all of the observations contained in its row.
- ☑ Attributes are used to label spreadsheet columns.
- ☑ A spreadsheet row is also referred to as an observation.
- ☑ An observation includes all of the attributes contained in its row.

4. A data team at a public university wants to communicate to stakeholders about which foreign language courses are most popular with students. They create a data visualization that identifies the 38 different courses, then shows the number of students enrolled in each one. What type of data visualization should they create?

- ○ Line chart
- ◉ Bar chart
- ○ Donut chart
- ○ Pie chart

5. When working in a spreadsheet, what is the correct syntax for a formula that adds the value in cell A1 to the value in cell B1?

- ◉ `=A1+B1`
- ○ `A1+B1=`
- ○ `=A1,B1`
- ○ `A1,B1`

6. Fill in the blank: The SQL clause `SELECT *` is used to _____ all data from a particular table.

- ○ copy
- ◉ retrieve
- ○ archive
- ○ exclude

7. In the following query, which clause indicates the table from which to retrieve data?

```
SELECT *
FROM Garden
WHERE herb = 'basil'
```

- ○ `WHERE herb`
- ◉ `FROM Garden`
- ○ `= 'basil'`
- ○ `SELECT *`

8. Which text wrapping feature cuts off the contents of a spreadsheet cell so only the text that fits is visible?

- ○ Fill
- ○ Wrap
- ◉ Clip
- ○ Overflow

9. Which of the following statements accurately describe data visualizations and visualization tools? Select all that apply.

- ☑ The process used for analyzing the data should not influence the type of visualization ultimately selected.
- ☐ When working with the programming language R, an integrated developer environment enables the creation of data visualizations.
- ☑ Spreadsheets have visualization tools that enable data professionals to create line or bar charts.
- ☑ Tableau makes it possible to integrate data into dashboard-style visualizations.

10. A data professional wants the data in column A to be in alphabetical order from A to Z. What spreadsheet feature would enable them to accomplish this task?

(n/a)	A	B	C
1	Deciani	Lin	33
2	Roberts	Jeremy	28
3	Storelli	Hyun	64
4	Piotrowski	Sook	84
5	Mondek	Priscilla	42
6	Lam	April	96

- ○ Sort range
- ○ Chart editor
- ◉ Formatting
- ○ Dragging