Data Visualization Toolkit

By Sophia Anderson | RJI Fellow 2025

Submit feedback on this toolkit here!

Toolkit Presentation
Python Troubleshooting Guide
Google Sheets Guide
Datawrapper Guide
Flourish Guide

Table of Contents

I need to make a visualization.

I need a template for a map.

Flourish:
Louisiana separated by counties
Louisiana Congressional Districts
New Orleans Public School District
New Orleans zip codes

To use, click *Duplicate & edit* in the top right.

I have a question about how to use Datawrapper, Flourish or Google Sheets.

FAQ guides for Datawrapper, sheets and Flourish.

I'm looking for good sources of data.

- Louisiana Dept, of Health Eat Safe Inspections: You can see restaurant inspections, including for kitchens at jails!
- Washington Post's climate change data set
- Google's Data Commons: A platform that hosts lots of data sets. You'll have to verify the accuracy of each data set but it's a good place to start looking.
- Harvard's Dataverse: This is a massive collection of academic/scientific research compiled by Harvard.
- French Institute for Demographic Studies
- Pew Research Datasets
- Sentencing Project data on people serving life sentences
- New Orleans Open Data - this is a spreadsheet of all the datasets they have
    - NOPD use of force
    - Reports of potholes in New Orleans: This one is a little funny, but residents can call 311 to report potholes, and there have been about 24,000 unique reports of potholes since 2019. This data set is all maintenance calls but you can sort to just see potholes.
    - Engagements between city council members and constituents

    - Schools and locations (updated each year)

Want to talk to a person? Find a list of experts for a given topic using this website: Rolli.

I have questions about using AI.

What is it good for?

- Reformatting finite data you give. For example, rearranging column headers on a big data set.
- Converting data from one format to another (see section on converting PDFs). Try pulling text from an image, pulling a transcript from a PDF and turning it into a text file, transcribing audio from videos, or converting something into a spreadsheet.
- Labelling data based on its location on a spreadsheet (see this example).

What is it NOT good for?

- Anything having to do with sensitive data. Don't give AI anything you wouldn't want the public to have access to (just in case).
- Finding information. Don't ask it for anything you don't already have access to or don't already know/can't verify. (Example: Give me a list of every school in New Orleans. Don't trust it to do that. It might not give you everything or give you schools that don't exist).
- Doing math problems. For some reason, it is prone to hallucinations when doing multiplication.

Some tips for maintaining accuracy:

- Always know how many data points you gave a model and make sure you gave the same amount back. For example, if you give it a list to sort, know how many things are on the list so you can check how long the list is when it gives it back.
- When you can, work with data in alphabetical order, so that if you're missing something, you know right where it went wrong because things are always in the same order
- Don't assume it knows what you're talking about if you're referencing information from earlier in a "conversation." Don't reference an earlier message. Instead, reference a file or resend a piece of data every time you have a question about the data.
- It is not always good at detecting duplicates in a list if some things in the list are very similar. For example, in a list of schools, if a middle and high school have the same name, it may not differentiate the two. It may think East Middle School and East High School are the same school, so double check.

Resources for more information on AI & journalism

- IRE25 talk “What is (and isn't) a good fit for AI-assisted journalism?” (audio) - Juliana Castro Varón, New York Times; Dylan Freedman, The New York Times
    - AI is pulling from content that has already been created, and therefore will never be original
    - You can use AI to sort through large amounts of information (like public records) and find the most relevant bits by asking it to rank documents by relevancy or finding key words
    - AI works well for converting information from one format to another
    - AI can find information on particular topics with more nuance than just looking for keywords (like finding related phrases)
- Derek Willis’s tips on using extractive AI
- Bloomberg’s tips for using AI to create a dataset from qualitative data

AI tools that aren’t just ChatGPT

- Claude AI - in addition to typical chatbot features, it has built-in games and workshops for learning skills like coding and languages
- Mistrall Small 3.1 AI - a French AI startup with a chatbot and API integration
- Groq (not the same as X’s grok!) - another AI chat bot
- Google AI Studio - Google’s AI chat bot, which is more customizable than ChatGPT and other similar software. It’s a “thinking” model, which just means it can explain the process behind its response if you want it to. This can help with troubleshooting.

Ethical and environmental concerns about AI

AI data centers use electricity and large amounts of water to cool down machinery. Data centers also take resources from low income residential areas, as seen with Meta’s data centers.

- Explained: Generative AI’s environmental impact - MIT News
- The Uneven Distribution of AI’s Environmental Impacts - Harvard Business Review
- Their Water Taps Ran Dry When Meta Built Next Door - NYT

If you’re using AI to parse data, you’re not at any risk of this, but it’s worth noting that some people have developed psychosis from becoming involved in parasocial relationships with AI chatbots.

- When the Chatbot Becomes the Crisis: Understanding AI-Induced Psychosis - Cognitive Behavior Institute
- They Asked an A.I. Chatbot Questions. The Answers Sent Them Spiraling. - NYT

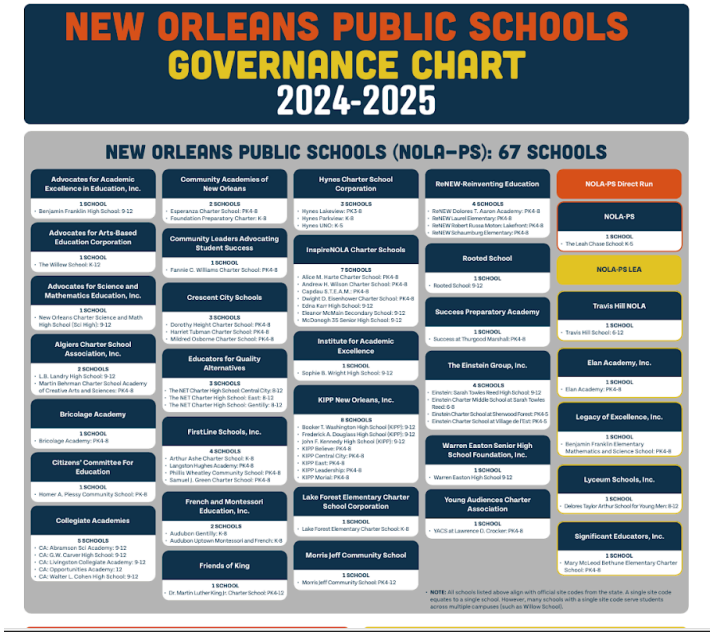
AI, even generative AI, is not creating anything new. It's essentially summarizing near infinite content that already exists on the internet. It is therefore always fallible and incredibly susceptible to human prejudices like racism and sexism.

- AI Hiring Tools Are Advising Women and Minorities to Ask for Lower Pay in Salary Negotiations - Inc.
- How AI reinforces gender bias—and what we can do about it - UN Women

I need to organize or analyze data.

I need to convert a PDF into something I can use.

Example: This is a PDF with a list of schools. It's the only place the schools are listed all together, but we need each school in its own cell in a spreadsheet so we can add more data. It also has extra information, so we can't just copy and paste all the text on the pdf into a new document.

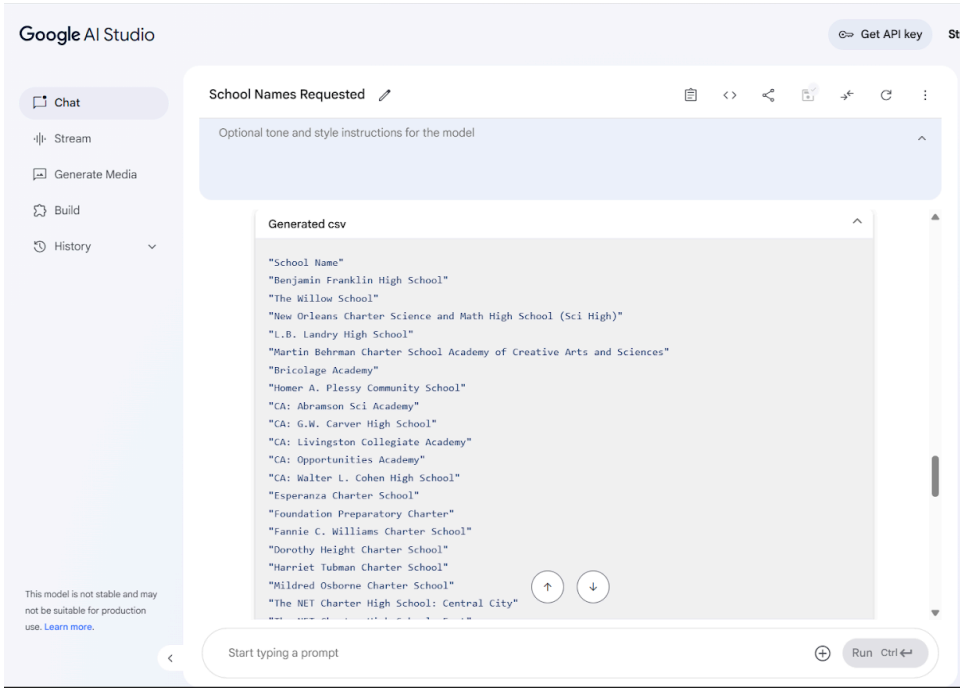


Option 1: Using LLMs to extract and organize the data

1. Upload the pdf to your LLM of choice (note: only do so with documents that don't contain sensitive information. This PDF is just school names and it's public record, so no issues there).
2. On the right hand side, set the Temperature to 0. This limits the creativity of the LLM.
3. Use the following prompt:

Please reply with only the names of schools in plain CSV format with absolutely no other text or response information. Use double quotes in all cases, no yapping.

4. Copy the text it gives you (it should look like this). Google AI can't create files for you to download outside of its internal system, which is why it can't just give you a CSV file to download, and you have to do a few extra steps.



5. Paste the text into a text editor like Notepad or TextEdit.
6. Choose save as, and title it "Title.CSV"
7. You can now open this data in an Excel or Google Sheets file and each school will be in its own cell.

Option 2: Use Python to scrape the information you need

1. If you don't have Python installed, this is a guide by IRE that teaches you how to install it. See this list of Python FAQs.
2. Install pdfminer.six from GitHub. Note that this has to be done in the command prompt, not in Python itself. If the line starts with >>>, you're in Python, not the command prompt. To open the command prompt, type cmd in the search bar of your computer.
3. Copy the following code into a text file:

```
from pdfminer.high_level import extract_text

pdf_file_path = "governance.pdf"
```

```
print(f"Attempting to read text from the file: {pdf_file_path}")

try:
    # This line calls the function from the library to read the PDF.
    extracted_text = extract_text(pdf_file_path)

    # These lines print out the text that was found.
    print("\n--- START OF EXTRACTED TEXT ---")
    print(extracted_text)
    print("--- END OF EXTRACTED TEXT ---\n")

    print("Successfully extracted text from the PDF!")

except FileNotFoundError:
    # This part runs only if the file 'governance.pdf' can't be found.
    print(f"Error: The file '{pdf_file_path}' was not found.")
    print("Please make sure the PDF file is in the same folder as the script.")
```

The ONLY thing you need to change in this code to personalize it, is to change where it says “governance.pdf” to the name of your pdf, so it looks like:

```
pdf_file_path = "mypdf.pdf"
```

4. Choose “save as” and save it as “read_pdf.py”
5. You’re going to need your code (this .py file you just saved), the pdf you’re scraping, and Python itself all in the same folder on your computer. To find out where your Python is stored, type where python into the command prompt and hit enter. The text that comes up is the path to where Python is stored. I’ll look something like this:

```
C: \Users\your name\something
```

6. Move your pdf and .py file to this place too.
7. Now, in the command prompt, enter this code:

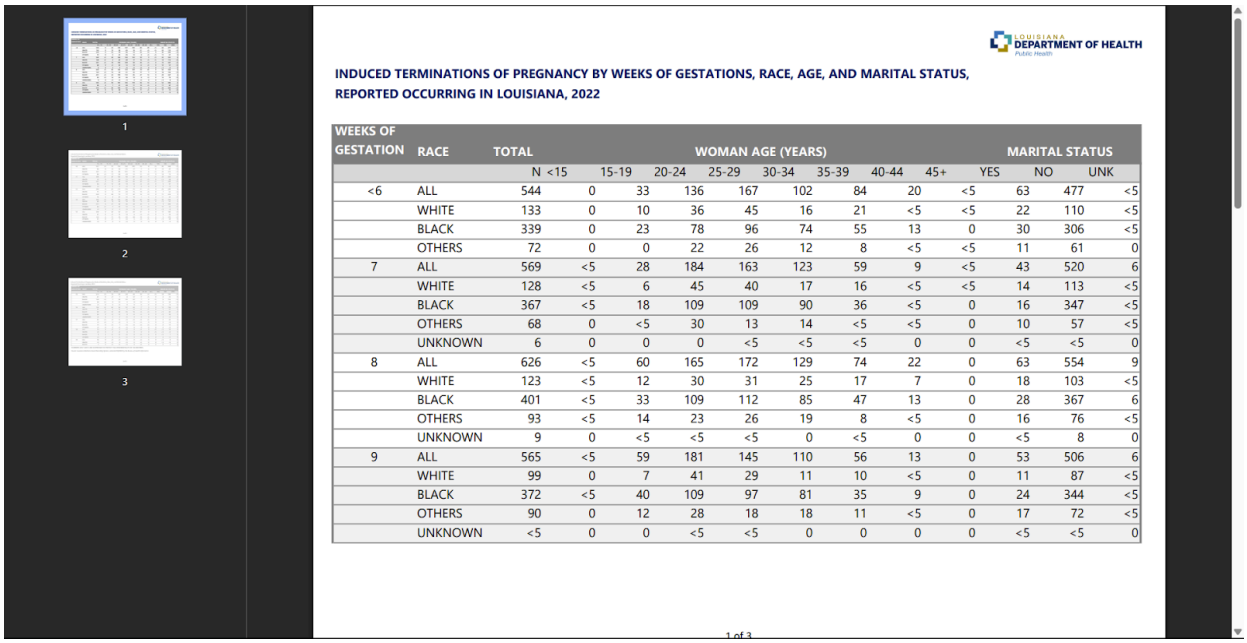
```
python read_pdf.py
```

This is going to fetch and run the code from above and then extract the text from your pdf. It should spit out all the text and you can copy and paste it wherever you need.

Option 3: Edit and convert in Adobe Acrobat

If you have access to Adobe Acrobat, you can use it to instantly convert pdfs into Excel files. This works best for files that are already formatted in a relatively organized way, but would be difficult to copy and paste into an Excel file. It doesn't work well for pdfs that have lots of extraneous graphics or text in no particular format.

Good example:



LOUISIANA DEPARTMENT OF HEALTH
Public Health

INDUCED TERMINATIONS OF PREGNANCY BY WEEKS OF GESTATIONS, RACE, AGE, AND MARITAL STATUS, REPORTED OCCURRING IN LOUISIANA, 2022

WEEKS OF GESTATION	RACE	TOTAL	WOMAN AGE (YEARS)							MARITAL STATUS		
		N	<15	15-19	20-24	25-29	30-34	35-39	40-44	45+	YES	NO	UNK
<6	ALL	544	0	33	136	167	102	84	20	<5	63	477	<5
	WHITE	133	0	10	36	45	16	21	<5	<5	22	110	<5
	BLACK	339	0	23	78	96	74	55	13	0	30	306	<5
	OTHERS	72	0	0	22	26	12	8	<5	<5	11	61	0
7	ALL	569	<5	28	184	163	123	59	9	<5	43	520	6
	WHITE	128	<5	6	45	40	17	16	<5	<5	14	113	<5
	BLACK	367	<5	18	109	109	90	36	<5	0	16	347	<5
	OTHERS	68	0	<5	30	13	14	<5	<5	0	10	57	<5
	UNKNOWN	6	0	0	0	<5	<5	<5	0	0	<5	<5	0
8	ALL	626	<5	60	165	172	129	74	22	0	63	554	9
	WHITE	123	<5	12	30	31	25	17	7	0	18	103	<5
	BLACK	401	<5	33	109	112	85	47	13	0	28	367	6
	OTHERS	93	<5	14	23	26	19	8	<5	0	16	76	<5
	UNKNOWN	9	0	<5	<5	<5	0	<5	0	0	<5	8	0
9	ALL	565	<5	59	181	145	110	56	13	0	53	506	6
	WHITE	99	0	7	41	29	11	10	<5	0	11	87	<5
	BLACK	372	<5	40	109	97	81	35	9	0	24	344	<5
	OTHERS	90	0	12	28	18	18	11	<5	0	17	72	<5
	UNKNOWN	<5	0	0	<5	<5	0	0	0	0	<5	<5	0

1 of 3

If you have a pdf that doesn't allow you to select text to copy and paste, using the "edit" feature will allow you to copy and paste individual blocks of text into a new document or spreadsheet. It's still more labor intensive than scraping the document, but works if you want to have a lot of control over the data and where it goes.

Option 4: Use Tabula to extract data

Tabula is a downloadable program that extracts data from pdfs.

1. Download Tabula
2. Download Java if you're on a Windows PC or using Linux.
3. Open Tabula and import a pdf
4. Using Tabula's interface, select the data to extract and export

I need to get rid of duplicates in my data.

Option 1: Use Google Sheets' built in filter

1. Go to Data >> Data Cleanup >> Remove duplicates
2. Use this guide

Option 2: Use Compare Sheets extension

1. Install this extension for Google Sheets
2. Use this guide

Option 3: Use Python to find duplicate rows

1. If you don't have Python installed, this is a guide by IRE that teaches you how to install it.
2. Make sure you're in your virtual environment by typing the following code. This guide by IRE explains virtual environments.

```
venv\Scripts\activate
```

3. Install pandas, a software library that helps with data manipulation. Do this by opening the command, and typing the following code:

```
pip install pandas
```

4. Save the spreadsheet you're examining as a CSV with underscores instead of spaces in the title. Save this spreadsheet to the location where you have Python and your virtual environment saved.
5. Open a text file and put in the following code:

```
import pandas as pd

df = pd.read_CSV('my_data.CSV')

duplicate_rows = df[df.duplicated()]

print("Found these duplicate rows:")
print(duplicate_rows)
```

The ONLY thing you need to change in this code to personalize it, is to change where it says 'my_data.CSVf' to the name of your spreadsheet, so it looks like:

```
df = pd.read_CSV('spreadsheet.CSV')
```

6. Save this text file to the same location as your spreadsheet under the title "find_duplicates.py".
7. Enter the following code into the command:

```
python find_duplicates.py
```

8. This will return a list of the duplicate rows in your spreadsheet. If there are no duplicates, it will return a list of the headers of each column in your spreadsheet.

If all the rows in your data are technically unique, but you want to find duplicates based on a combination of two factors, here's how. (For example, if a row has the same location AND the same time, it's considered a duplicate).

1. Find the line of code in your text file that looks like this:

```
duplicate_rows = df[df.duplicated()]
```

And replace it with this:

```
duplicate_rows = df[df.duplicated(subset=['Location', 'Date Created'])]
```

In this case, the customization here is finding rows that are identical based on a combination of location and date created. You can replace these words with the title of the columns you want to use.

2. Save your text file and run the following code again:

```
python find_duplicates.py
```

3. The list it returns will show you the second, third, fourth, etc. occurrence of a duplicate line, but not the original. To have Python show you the original line, and save all the duplicate lines to a new file, replace all the previous code in your text tile with the following:

```
import pandas as pd

file_name = 'my_data.CSV'

subset_columns = ['Location', 'Date Created']

output_file_name = 'duplicate_report.CSV'

df = pd.read_CSV(file_name)

all_duplicates = df[df.duplicated(subset=subset_columns, keep=False)]

all_duplicates.to_CSV(output_file_name, index=False)

print(f"Found {len(all_duplicates)} total duplicate rows.")
print(f"Saved the full report to: {output_file_name}")
```

Remember to substitute in the name of your spreadsheet in the second line, and the column names you're checking for in the third line, where it says Location and Date Created.

4. Save your text file and run the following code:

```
python find_duplicates.py
```

You should now have a spreadsheet titled Duplicate Report saved to the same folder as all your other documents for this project.

I need to check the limitations of my data.

Here are some common things to check for in data sets to make sure you understand the potential limitations of your data.

- Are certain values suppressed? Sometimes data sets won't include numbers for demographics that are too small, for privacy reasons.
- Is the data preliminary or final? Is there any data that is "projected" instead of absolute?
- Have any of the values in the data set already been rounded?
- Does your data include a margin of error or an accuracy estimate?

- Does your data come with a dictionary/index to define terms?
- When was it published? Does it have an “as of” date?
- How do the values correspond to dates? For example, are the monthly values the total per month, or the total for the last year up to that month? (see below)

I need to turn year-to-date data into monthly data.

Sometimes you’ll get data that’s separated by month, but the value for each month represents the last twelve months (or TTM, trailing twelve months). This means that the data for June, isn’t just the data for June, it’s for the twelve months leading up to June. If you need to show data broken down by month, this doesn’t work. Here’s the best way to break it down.

To find the value for a given month, you need to subtract the TTM for the previous month and then add the value of the month from one year ago.

So to find June 2025:

(TTM June 2025 - TTM May 2025) + June 2024

The issue is that if you’re working with TTM values, you likely won’t have the June 2024 value, as that’s the issue we’re trying to solve. To fix this, we can estimate a seed value by finding the average per month for 2024. Add all the months in 2024 and divide by 12. Then assign that approximate value as the seed value for each month in 2024.

Using that seed value, you can now calculate the individual monthly value for June 2025.

Look at the table below for an example.

Month	TTM value	Monthly value	
Jan 2020	201	22.25	These are the seed values calculated by finding the average monthly value for all the months in 2020.
Feb 2020	207	22.25	
March 2020	201	22.25	
April 2020	208	22.25	
May 2020	225	22.25	
June 2020	227	22.25	
July 2020	228	22.25	

August 2020	228	22.25	
September 2020	236	22.25	
Oct 2020	239	22.25	
Nov 2020	249	22.25	
Dec 2020	267	22.25	
Jan 2021	278	25.25	
Feb 2021	285	28.25	This is 285 - 278 + 22.5
March 2021	306	39.25	
April 2021	313	41.25	
May 2021	309	18.25	
June 2021	314	27.25	
July 2021	323	31.25	
Aug 2021	329	28.25	
Sep 2021	343	36.25	
Oct 2021	357	36.25	
Nov 2021	349	14.25	
Dec 2021	348	21.25	
Jan 2022	341	18.25	
Feb 2022	347	34.25	
March 2022	330	22.25	

I need to standardize messy data.

Use *Find and replace* feature to mass replace data that's not formatted how you want it.

1. Ctrl + f on Windows (or Cmd + f on Mac) opens a search bar in the top right
2. Click on the three dots on the right next to the X
3. Put the data you're looking for in *Find*
4. Put what you want to replace it with in *Replace with*
5. If you only want to apply this within a certain part of your data, click on the dropdown that says *All sheets* and select *Specific range*. Put in the range of cells you want to search
6. Note if you want your data to be case sensitive
7. Click *Replace all*

Google Sheets Guide for how to resize rows and columns, find empty cells, remove duplicates, sort columns, and more.

I'm working with geographic data.

I have coordinates for a polygon, but there are too many.

If you're defining a shape in a map, like the outline of a county or a zip code, it will be formatted as a polygon or multipolygon. This just means that there are hundreds of points that connect to make up the outline of an area. For large or complicated shapes, there might be more points than your spreadsheet or map can handle. In the typical spreadsheet, the character limit for a cell is 32,767 characters, and the column limit is 16,384 columns.

GeoJSON files are formatted as one long line, so in a spreadsheet, it will either fill in one cell, or put one coordinate per column. This means that if you have more than 16,384 coordinates, your polygon won't fit in a spreadsheet. You can't separate the coordinates into different rows and label them all as the same region because they won't properly connect and it will look funky. The solution is to simplify the coordinates that make up your shape. You can do this in Python.

1. Save your coordinates into a text file as "name.txt." They have to be formatted correctly for this to work.
2. Open the Python command by typing cmd into your computer's search bar/menu. To install a package that helps work with geometric files, run this line of code:

```
pip install shapely
```

3. Then you're going to create another text file containing your code. This is the code:

```
from shapely.wkt import loads
from shapely.geometry import MultiPolygon, Polygon

def extract_coords(geometry):

    all_coords = []
    if geometry.geom_type == 'Polygon':
        # Get coordinates from the exterior boundary
        all_coords.extend(list(geometry.exterior.coords))
        # Get coordinates from any interior boundaries (holes)
```

```
        for interior in geometry.interiors:
            all_coords.extend(list(interior.coords))
    elif geometry.geom_type == 'MultiPolygon':
        # Loop through each polygon in the multipolygon
        for poly in geometry.geoms:
            all_coords.extend(list(poly.exterior.coords))
            for interior in poly.interiors:
                all_coords.extend(list(interior.coords))
    return all_coords

input_filename = 'name.txt'

try:
    with open(input_filename, 'r') as f:
        wkt_string = f.read().strip()
except FileNotFoundError:
    print(f"Error: '{input_filename}' not found. Make sure it's in the same folder as the script.")
    exit()

try:
    original_geometry = loads(wkt_string)
except Exception as e:
    print(f"Error parsing the WKT string: {e}")
    print("Please make sure the text file contains the correct MULTIPOLYGON string.")
    exit()

print(f"Successfully loaded the geometry.")
print(f"Original object type: {original_geometry.geom_type}")

tolerance = 0.001
simplified_geometry = original_geometry.simplify(tolerance, preserve_topology=True)

print(f"\nSimplified with tolerance={tolerance}")
print(f"Simplified object type: {simplified_geometry.geom_type}")

all_coords = extract_coords(simplified_geometry)

formatted_coords = ",".join([f"[{lon},{lat}]" for lon, lat in all_coords])
```

```
print("\nSimplified coordinates in your custom format:")
print(formatted_coords)

output_filename = 'simplified.txt'
with open(output_filename, 'w') as f:
    f.write(formatted_coords)

print(f"\nResults have been saved to '{output_filename}'")
```

The only thing you need to change is to make sure that 'name.txt.' is the name of your file with your coordinates in it. This script will simplify the coordinates of your shape and return the new coordinates in a file called simplified.txt. Name this file script.py.

4. Make sure this script file and your coordinate file are saved in the same place on your computer as your Python installation.
5. Run the script by typing

```
python script.py
```

I need to get coordinates for an address or vice versa

Option 1: Use Google Earth

- You can start a new project in Google Earth, plug in any address, and save it to your project
- Addresses automatically save with latitude and longitude
- This is helpful if you're trying to convert lots of addresses for one project all at the same time AND you want to be able to go back and look at them on a map
- You can export all the addresses at once by clicking *export project as KML*
- You'll then have to convert the KML file to something you can more easily upload to a map making software, like a CSV file or GEOjson coordinates
    - You can use mygeodata, convert to CSV, honeycomb maps, or quickmaptools (all free under a certain limit)

Option 2: Use Geocodio

- You can upload an entire spreadsheet of addresses to Geocodio and it will give you back a spreadsheet with all the addresses converted to latitude and longitude

- You'll just have to answer some questions about how your spreadsheet is formatted
- This tool is free as long as you stay under the limit of 2,500 conversions per day

Option 3: Use Datawrapper's built-ins

- Datawrapper has most region outlines that you could want for a map (like counties, zip codes, school districts, etc. within a given state)

I need to turn map data into something I can use

I need help with Python.

See this FAQ & common problems guide.

I need to check if my file is properly formatted.

When you're working with a coding software that's super picky, like GitHub, a file with a small formatting error can cause your whole project to fail. To check if your file is correctly formatted, use these tools:

JSON validator
geoJSON validator
Python syntax checker
CSV validator
HTML validator
HTML preview – allows you to instantly see what your HTML generates
XML validator for checking KMLs – KML files are types of XML files

I need to publish a visualization.

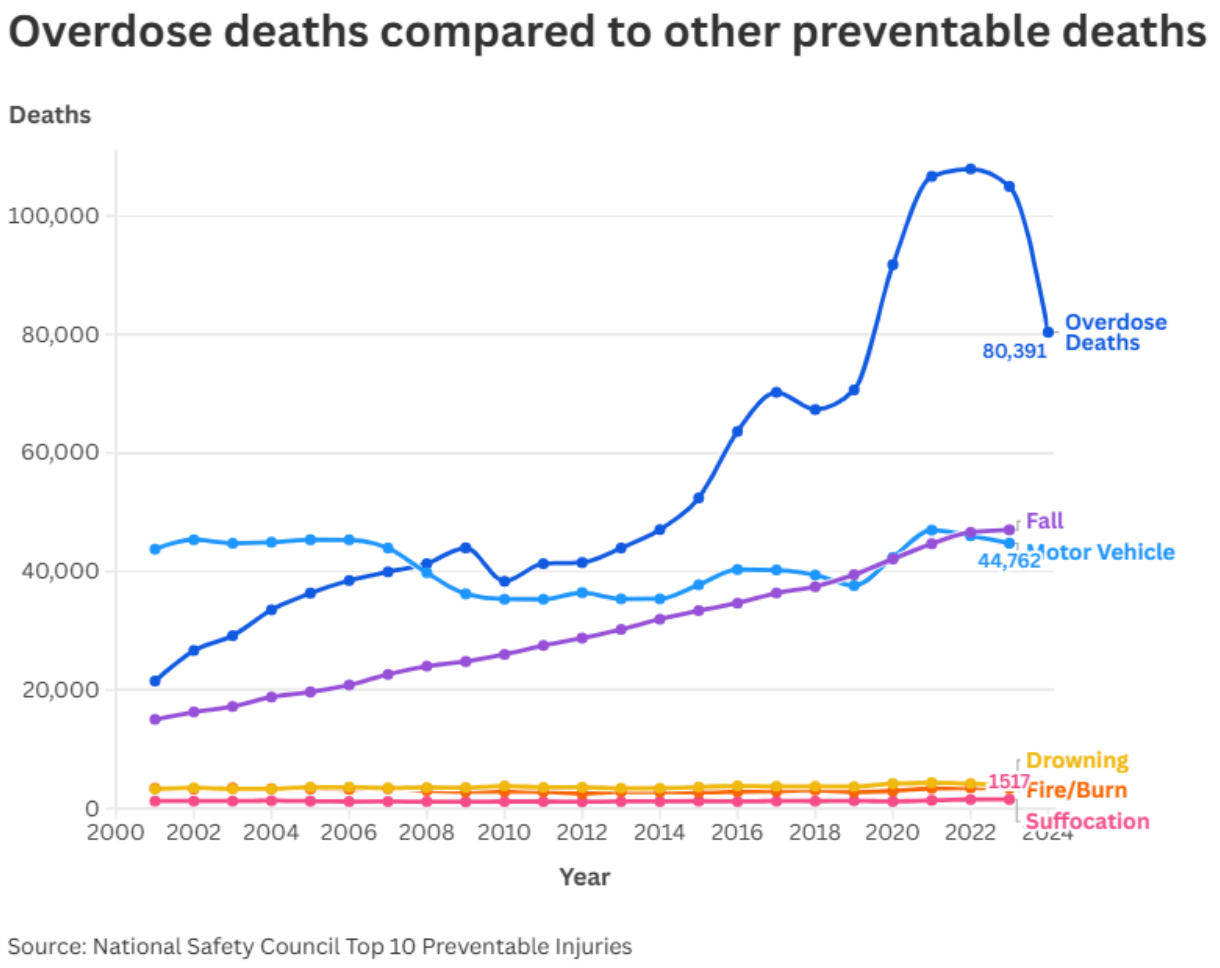
1. Click *Export and Publish* in the top right (Flourish). Copy the code under where it says *Embed on your website.*
2. Open the story you're working with in Grove. Enter the body of the article.
3. Add a new module and choose *Container Module* as the type.
   Choose *Column 1* where it says *Rows.*
4. In *Column 1*, add *HTML Embed* and paste the code from your Flourish graphic.

5. Click *Save and close* and make sure it appears correctly.

I need to get a quick snapshot of trends in my data.

Some questions to ask yourself:

- Can I compare this data to the same data state-wide, nationally, or to another state with similar demographics?



Source: National Safety Council Top 10 Preventable Injuries

For example, to give context to overdose deaths, we compare it to the other top five causes of preventable deaths in the US.

- Can I split this data up by age, sex, race, economic class or other demographics?
- Can I spread this data out over more time (i.e. divide years into months) or condense it into less time (i.e. combine years into decades)?
- Are there duplicates I can eliminate? Is there any data that's definitely irrelevant that I can remove?
- Can I divide this data by geography?

I need to keep track of my data for future fact-checking.

- Always make a copy of the original spreadsheet and label it as a copy
- Always lock your header row so it doesn't move if you sort your rows
- Don't delete rows or columns in spreadsheets with lots of data if you can help it. Instead, black out the cells you don't want, or choose *delete cells* and *shift up* or *shift left*. This way you don't accidentally impact more rows than you mean to.
- If you're adding a row or column that includes values that are calculations based on other columns, make sure you indicate that in the column header. For example, indicate if a column is the result of any math, like an average or percentage. This way you can always double check your math. Even better, use a Sheets formula and have Sheets do the math for you.
- Export and label every version of your data so you can go back and identify which step you made a mistake if necessary. For example, if you upload a spreadsheet to Flourish and your map doesn't look right, correct the data in your Google Sheet, not in Flourish, and re-export the Sheet labelled version 2.
- Keep track of how you're rounding numbers, if you're rounding. Always round to the same place. In Sheets, check how your cells are formatted under *Format* > *Numbers* to make sure Sheets isn't automatically rounding numbers for you.
- Note any limitations in your data sources and pass those on to viewers/listeners (i.e. if the agency collecting your data rounded numbers, omitted figures for privacy, etc.).

I want to try a visualization tool that isn't Datawrapper or Flourish.

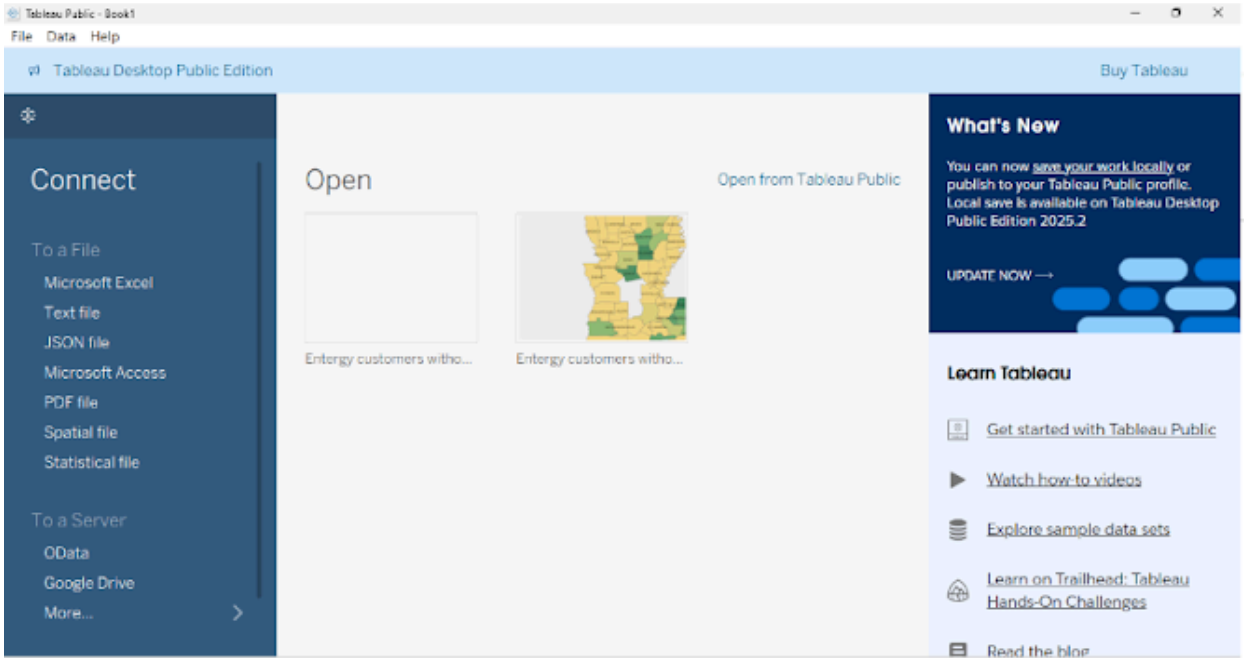
How about Tableau? Make sure you're using Tableau Public. You can download it to your computer for free, unlike the normal version of Tableau, which costs money after the free trial runs out.

Tableau's dashboard might look a little intimidating at first, but the basic workflow is separated into three steps. It can be as simple or as complicated as you want, depending on your skill level.

1. Upload your data set
2. Define the terms in your data
3. Create a visualization based on those terms

Step two is the most difficult and most important. If you don't define your data set correctly, it will be way too hard to create something with it.

Uploading your data set in Tableau

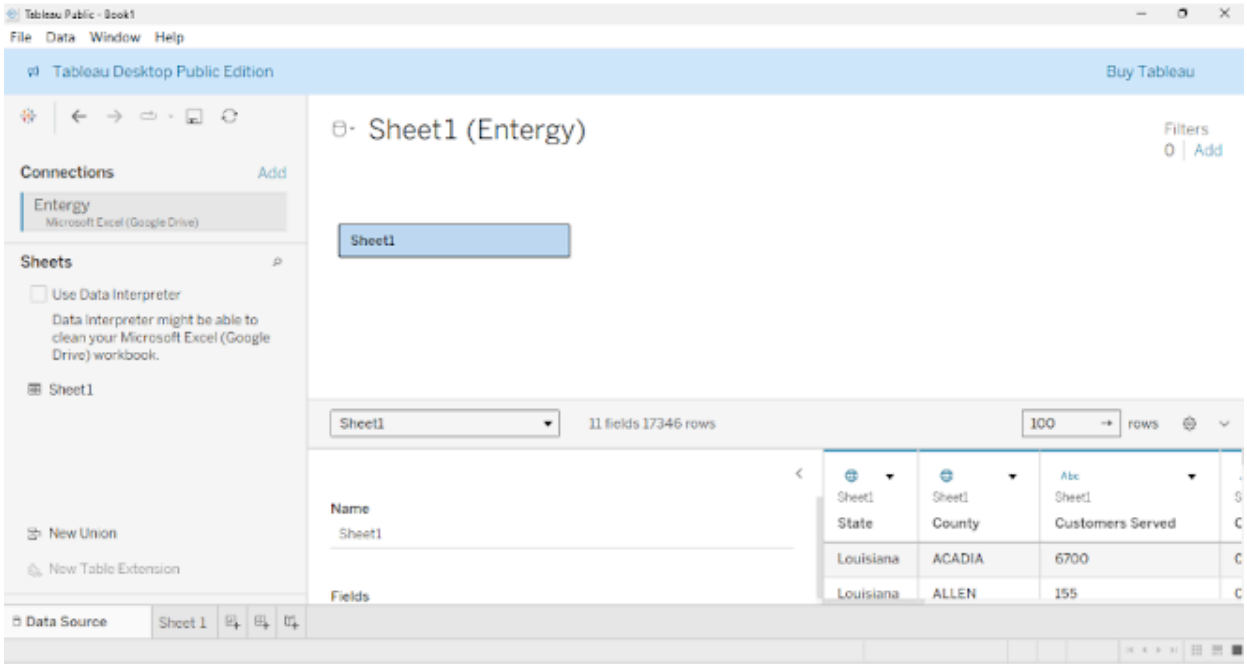


When you first open Tableau, this is what it looks like. If you've never created a visualization before, there won't be the option to open existing maps or charts. Start by connecting to your data.

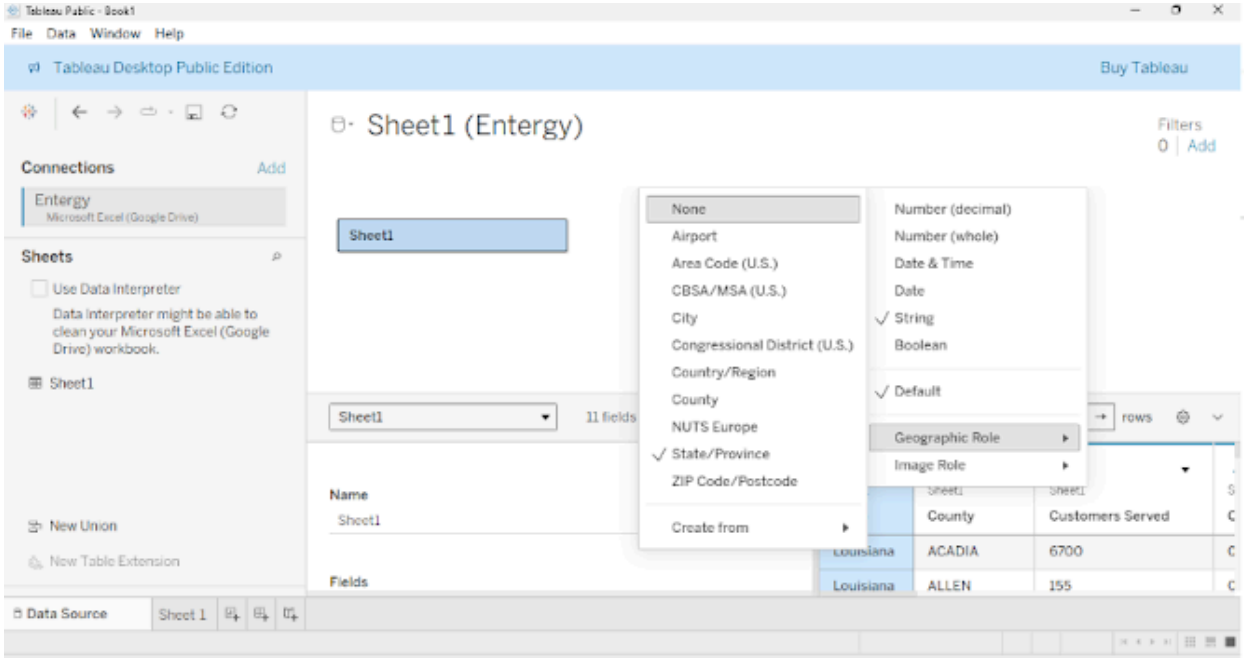
If you're working in a Google Sheet, either download it as a PDF or Excel file, or connect to your Google Drive. If you choose to connect to your Google Drive under *To a Server,* it will have you log in to your Google account and choose any spreadsheet in your Drive.

Editing the data in Tableau

Once you select a data source, this is the next screen you'll see. My sheet is called Entergy.



You can rename your sheet and any of the column titles by clicking on their titles.

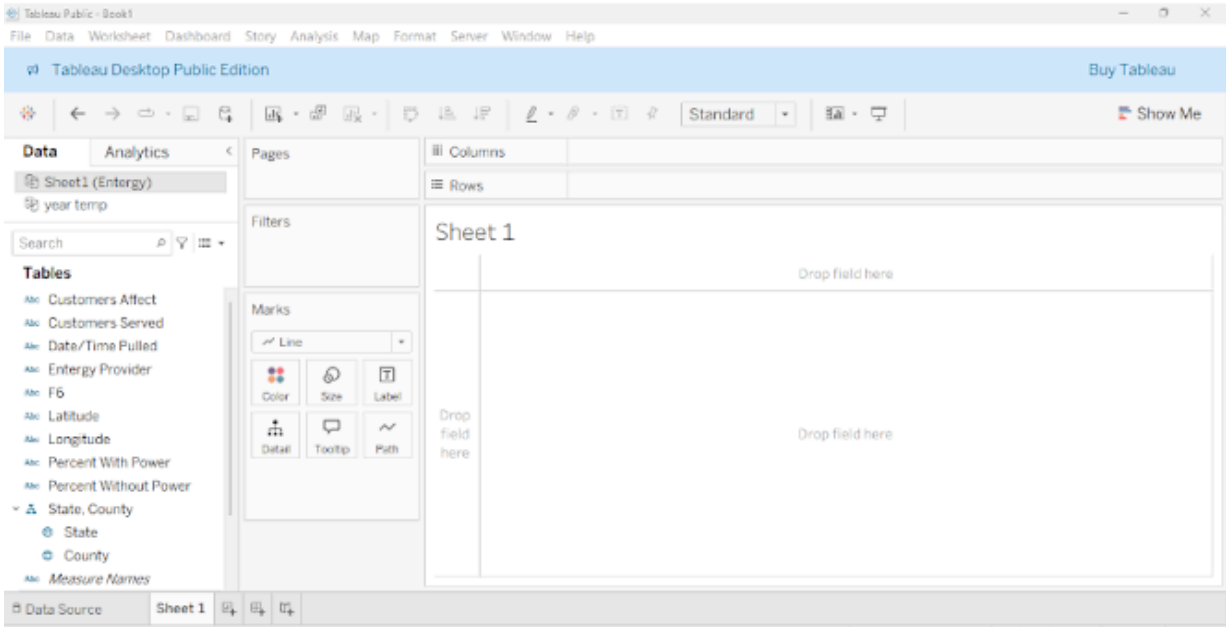


Click on the symbol above a column to change the type of data it's labelled as. This is especially important for geographic data.

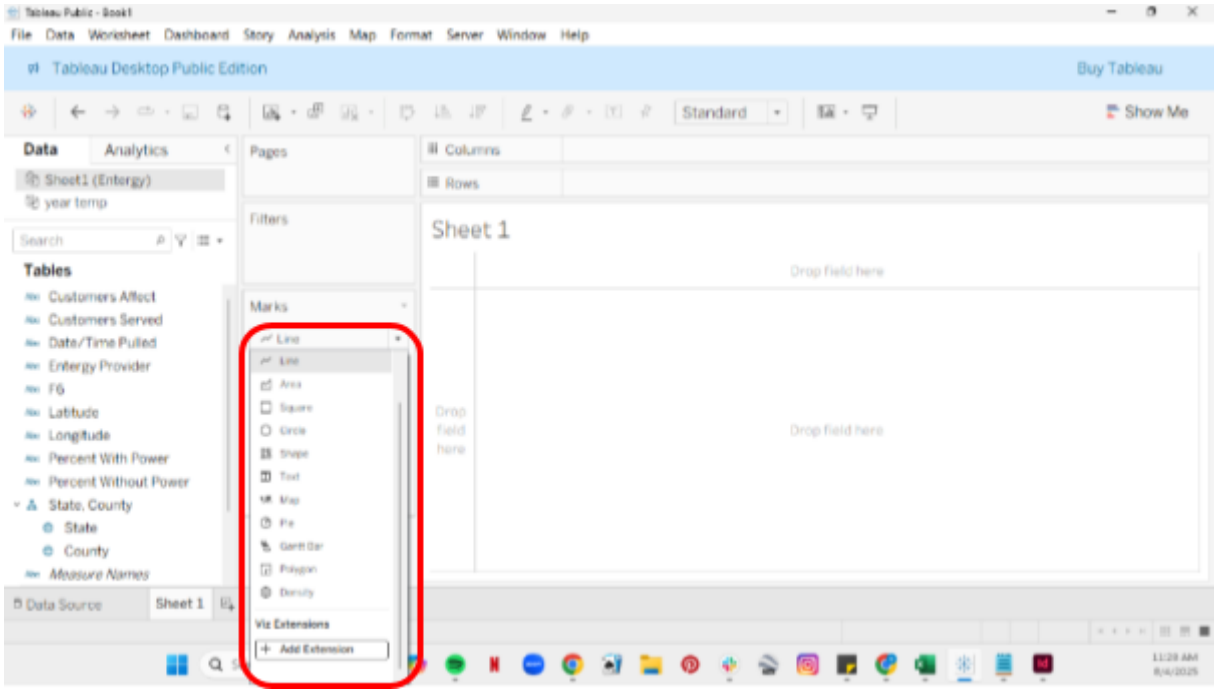
Almost any data that is a plain sequence of letters, numbers or characters will be classified as *String* data unless you choose otherwise.

Working in Tableau's worksheet

Click on *Sheet 1* in the bottom left to open up the editing dashboard, known as the worksheet. This is what it looks like:

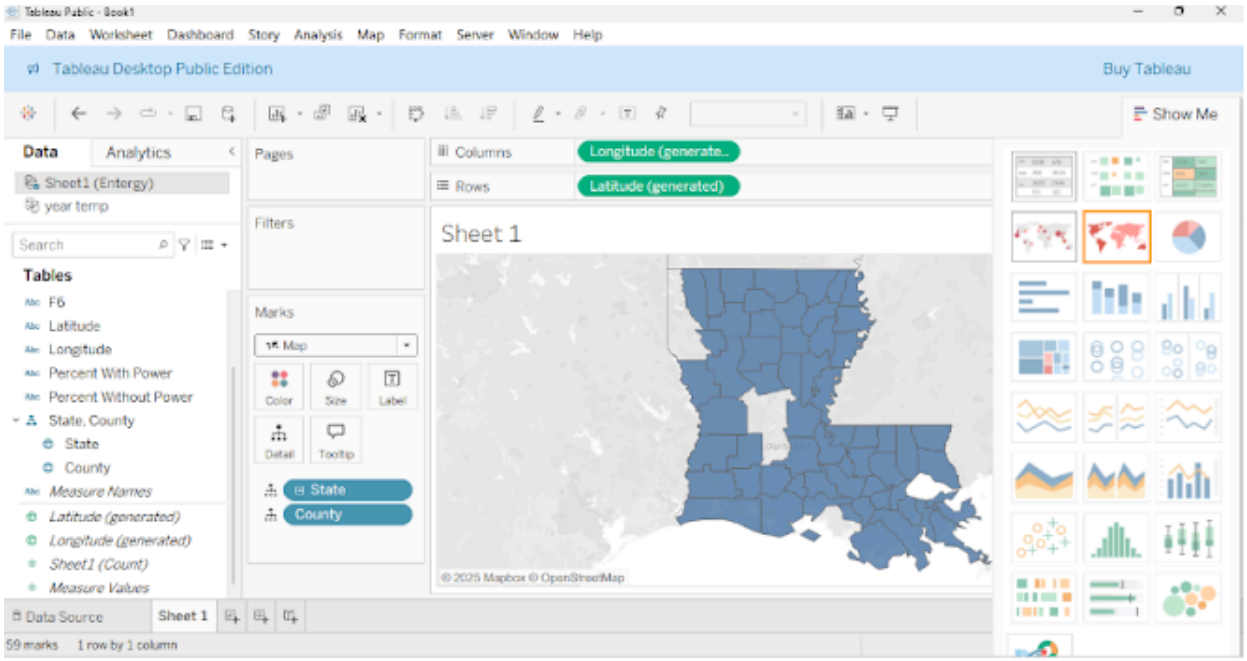


You can click and drag column names into the *columns* and *rows* fields, or into the *marks* boxes to choose which data determines color, size, labels, etc.



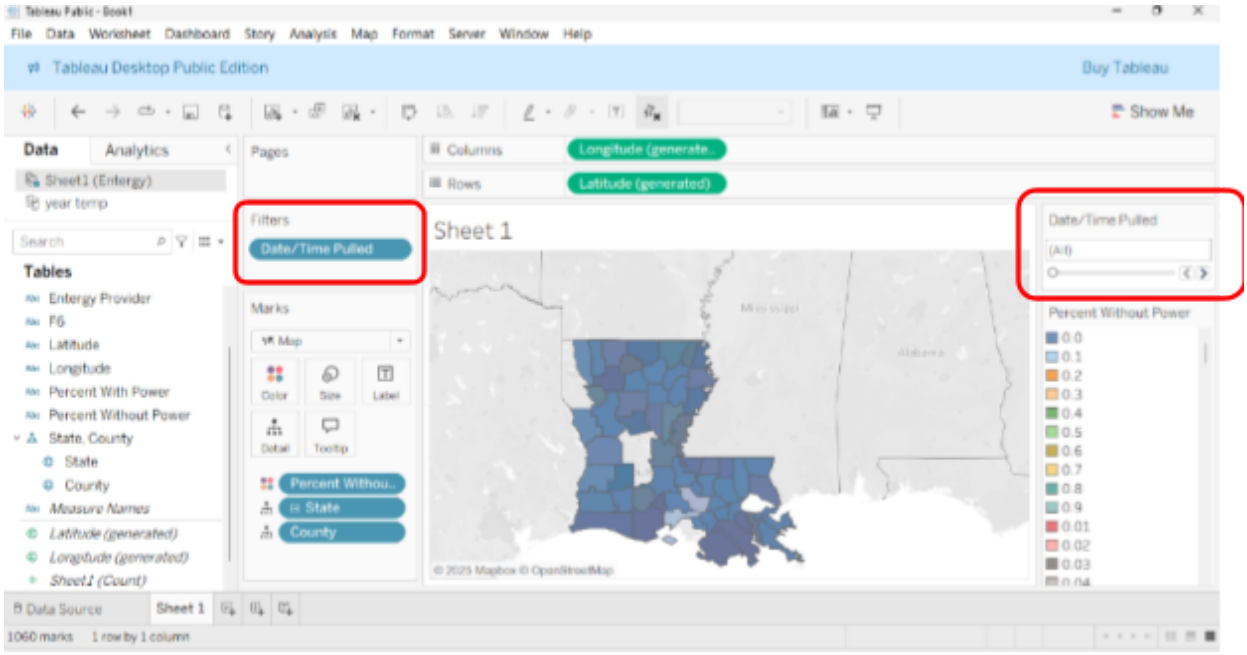
Click the drop down menu to choose what type of visualization you want. You can also start by dragging your data to the appropriate fields and Tableau will choose one for you. If it's not the kind you want, you can manually change it.

I put the longitude and latitude of my counties into the columns and rows fields, and then chose a chloropleth map under *Show Me* in the top right.



Customizing a visualization in Tableau

Since I want to color each county by how many customers are without power, I dragged Percent Without Power from the *Tables* column on the left to the box that says *Color.* Then, to filter by time, I dragged *Date/Time Pulled* into the *Filters* box. I right clicked on the *Date/Time Pulled* button and selected *Show filter*. Then I clicked on the filter that showed up on the right side and chose to display it as a single select slider.



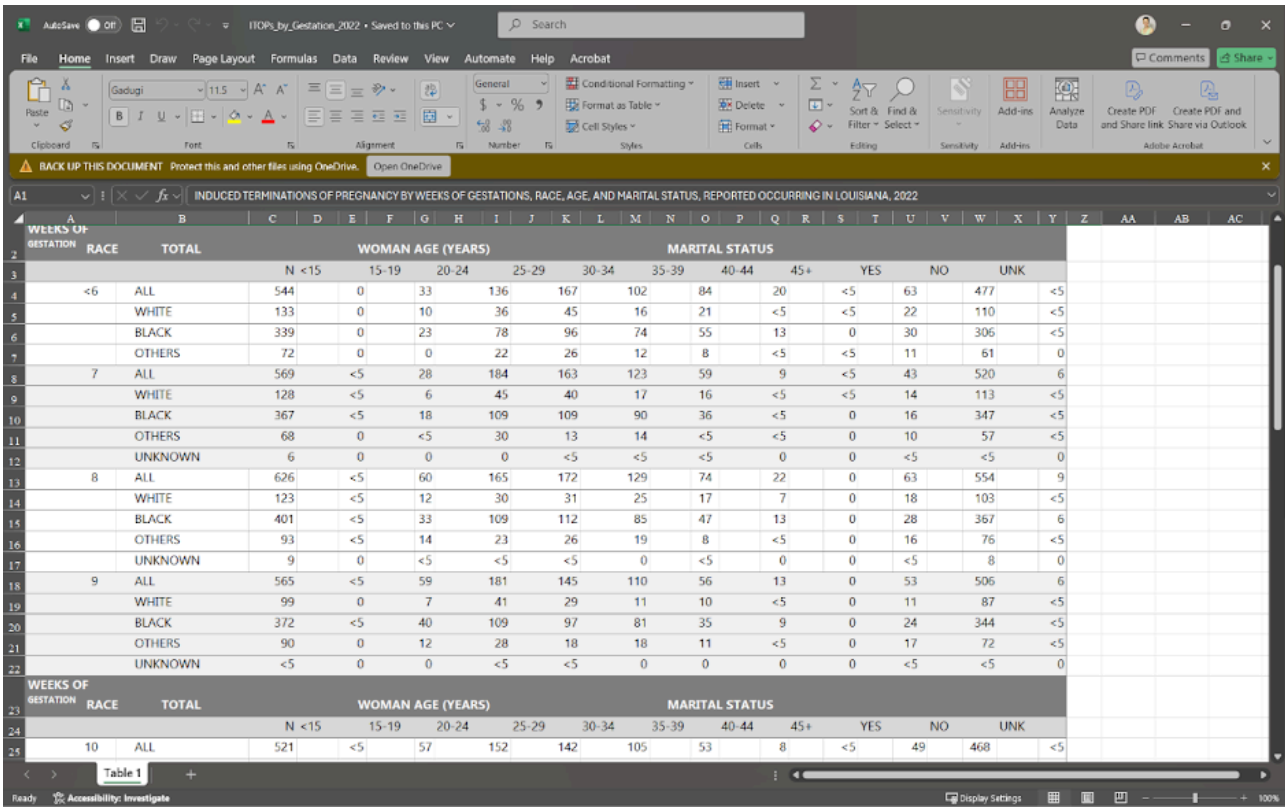
I added a label by dragging *County* to the label box. I customized the content that comes up when you hover over a county by clicking on the *Tooltip* box and typing in the column names that contained data I wanted to display.

You can publish a visualization by going to *File* > *Save to Tableau Public.*

How'd you do that? (examples)

Abortion data story graphics

1. Imported all the data into Google Sheets (my preference). This data and this data were already in CSV format, but the Louisiana Dept. of Health data was unfortunately in a PDF.
2. To get the DOH data into an editable format, I opened the PDF in Adobe Acrobat and converted it to an excel file. That helped, but some of the formatting was still a little weird.

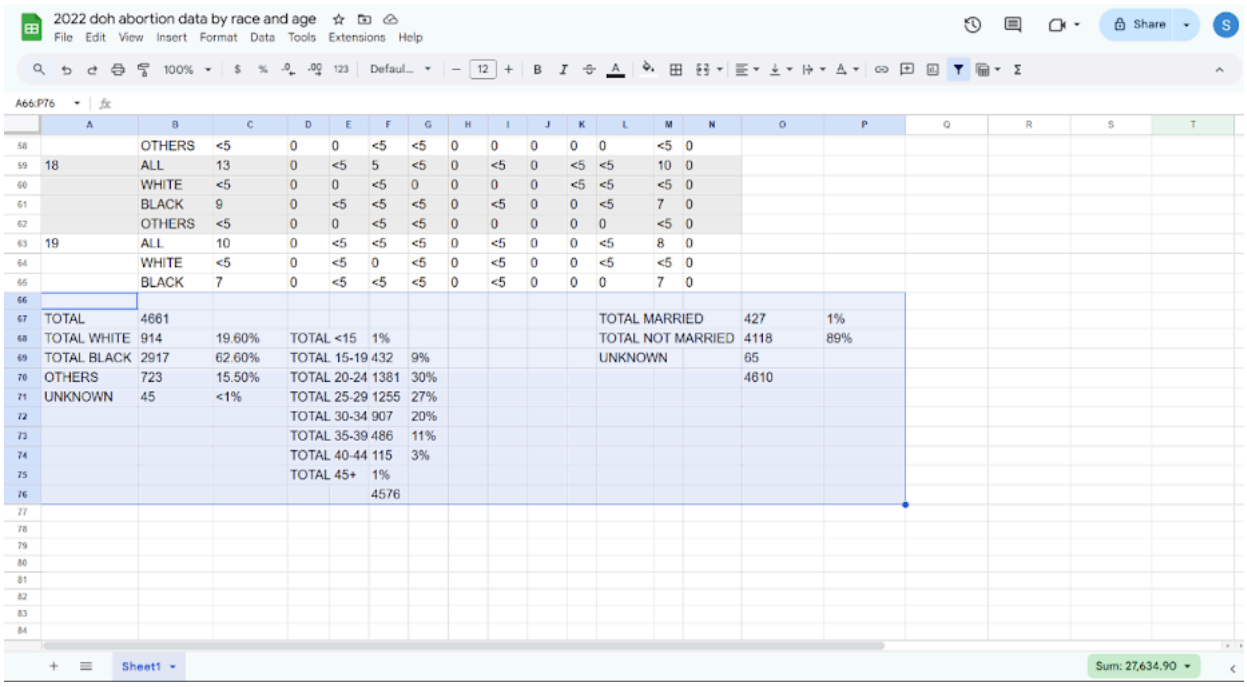


INDUCED TERMINATIONS OF PREGNANCY BY WEEKS OF GESTATIONS, RACE, AGE, AND MARITAL STATUS, REPORTED OCCURRING IN LOUISIANA, 2022

WEEKS OF GESTATION	RACE	TOTAL	WOMAN AGE (YEARS)							MARITAL STATUS		
		N	<15	15-19	20-24	25-29	30-34	35-39	40-44	45+	YES	NO	UNK
<6	ALL	544	0	33	136	167	102	84	20	<5	63	477	<5
	WHITE	133	0	10	36	45	16	21	<5	<5	22	110	<5
	BLACK	339	0	23	78	96	74	55	13	0	30	306	<5
	OTHERS	72	0	0	22	26	12	8	<5	<5	11	61	0
7	ALL	569	<5	28	184	163	123	59	9	<5	43	520	6
	WHITE	128	<5	6	45	40	17	16	<5	<5	14	113	<5
	BLACK	367	<5	18	109	109	90	36	<5	0	16	347	<5
	OTHERS	68	0	<5	30	13	14	<5	<5	0	10	57	<5
	UNKNOWN	6	0	0	0	<5	<5	<5	0	0	<5	<5	0
8	ALL	626	<5	60	165	172	129	74	22	0	63	554	9
	WHITE	123	<5	12	30	31	25	17	7	0	18	103	<5
	BLACK	401	<5	33	109	112	85	47	13	0	28	367	6
	OTHERS	93	<5	14	23	26	19	8	<5	0	16	76	<5
	UNKNOWN	9	0	<5	<5	<5	0	<5	0	0	<5	8	0
9	ALL	565	<5	59	181	145	110	56	13	0	53	506	6
	WHITE	99	0	7	41	29	11	10	<5	0	11	87	<5
	BLACK	372	<5	40	109	97	81	35	9	0	24	344	<5
	OTHERS	90	0	12	28	18	18	11	<5	0	17	72	<5
	UNKNOWN	<5	0	0	<5	<5	0	0	0	0	<5	<5	0
WEEKS OF GESTATION	RACE	TOTAL	WOMAN AGE (YEARS)							MARITAL STATUS		
		N	<15	15-19	20-24	25-29	30-34	35-39	40-44	45+	YES	NO	UNK
10	ALL	521	<5	57	152	142	105	53	8	<5	49	468	<5

I imported the excel file into Google Drive and edited it in Sheets until it looked better and the columns were standardized. (see here)

3. Once I had all the data to work with, Rosie and I decided which data was the most important to show instead of describing. We decided that the quantity of abortions over time, and the breakdown of abortions by gestation and race were top priority. We also wanted people to understand how abortions were impacting people around them, so we decided to make a visualization that showed the impact on different ages as a proportion out of 100.
4. To make this graphic, I used the data from 2021 as the sample, because it's the last year we have complete data from due to the full ban in 2022. I calculated the percentage of people in each age group. The percentages translate to how many people out of 100 would be in that age group.



2022 doh abortion data by race and age

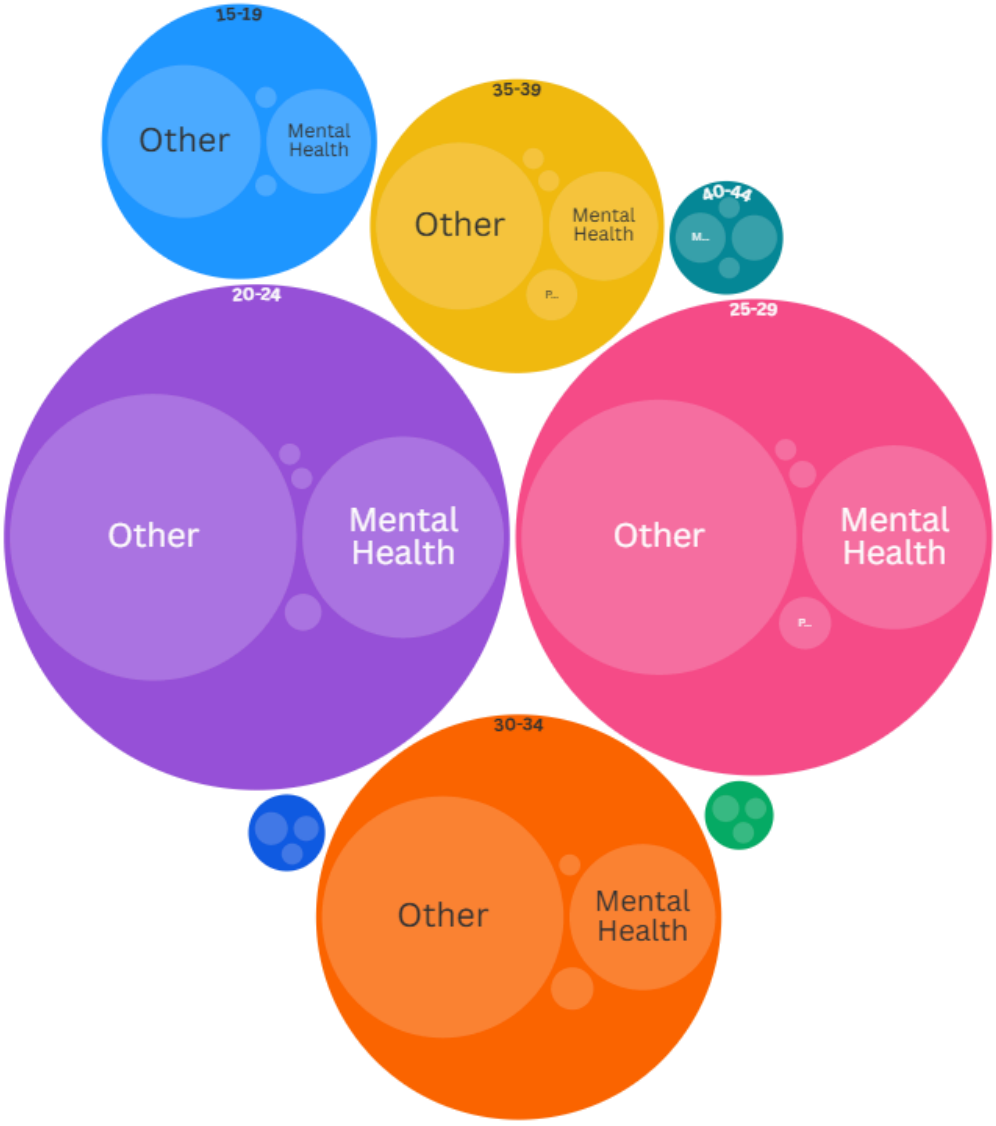
		OTHERS	<5	0	0	<5	<5	0	0	0	0	0	<5	0						
	18	ALL	13	0	<5	5	<5	0	<5	0	<5	<5	10	0						
		WHITE	<5	0	0	<5	0	0	0	0	<5	<5	<5	0						
		BLACK	9	0	<5	<5	<5	0	<5	0	0	<5	7	0						
		OTHERS	<5	0	0	<5	<5	0	0	0	0	0	<5	0						
	19	ALL	10	0	<5	<5	<5	0	<5	0	0	<5	8	0						
		WHITE	<5	0	<5	0	<5	0	<5	0	0	<5	<5	0						
		BLACK	7	0	<5	<5	<5	0	<5	0	0	0	7	0						
	TOTAL	4661										TOTAL MARRIED			427	1%				
	TOTAL WHITE	914	19.60%	TOTAL <15		1%						TOTAL NOT MARRIED			4118	89%				
	TOTAL BLACK	2917	62.60%	TOTAL 15-19		432	9%					UNKNOWN			65					
	OTHERS	723	15.50%	TOTAL 20-24		1381	30%								4610					
	UNKNOWN	45	<1%	TOTAL 25-29		1255	27%													
				TOTAL 30-34		907	20%													
				TOTAL 35-39		486	11%													
				TOTAL 40-44		115	3%													
				TOTAL 45+		1%														
						4576														

Sheet1 Sum: 27,634.90

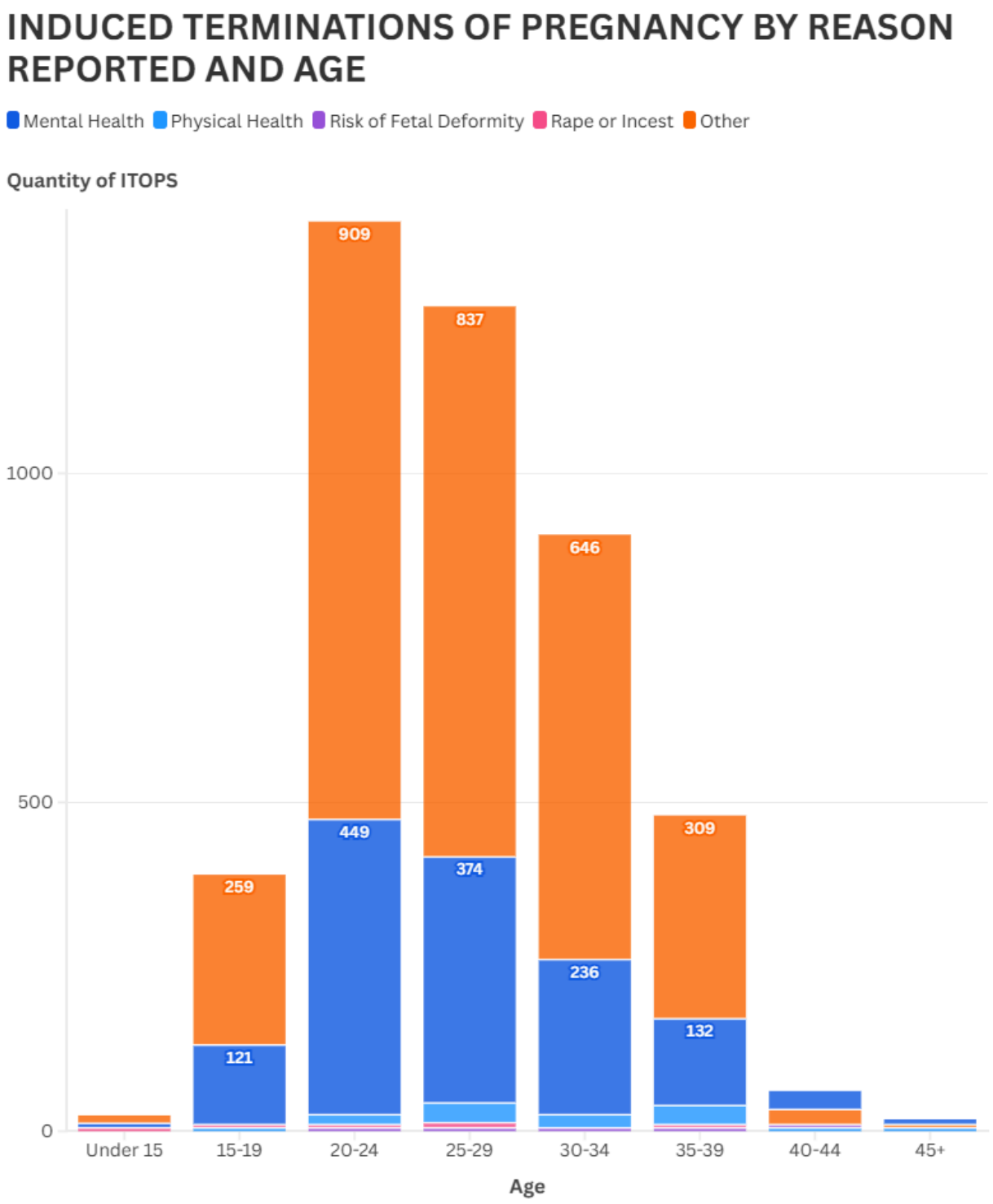
5. The other graphics were bar charts, and were relatively straightforward to make. The most customization we did was creating a color palette to standardize all the graphics.
6. I originally wanted to make a visualization using data from the DOH on the reasons people gave for getting abortions. But I ran into an issue with scale quickly, because most people put their reason as “other,” and this overshadowed the differences between the smaller categories. Below are two different ways I tried to show the data. I eventually decided not to visualize this data because there was no clear way to show it.

Abortions by reason reported and age (2022)

All



Source: Louisiana Electronic Event Recording System, extracted 08/2023 by the Bureau of Health Informatics • Numbers lower than 5 were suppressed to protect anonymity in the original study so values = "<5" were rounded to 5 in this graphic.



7. I added footnotes to all of the graphics, noting that some of the data sets repressed values below a certain number to protect the privacy of respondents. This means that some of the values are slightly smaller than in actuality.

Standardizing tons of school directories

To make a map of schools in New Orleans and where they've moved since 2000, I had to use directories provided by the school district. But they change slightly every year, so I couldn't just copy and paste the data from every school year into one spreadsheet under the same column headers. Here's how I fixed it:

1. Decide on the information I need, since there were some irrelevant columns. This was school names, school year, address and grade levels.
2. Using Google AI, I copied and pasted the entire spreadsheet from each school year, one at a time, and gave it this prompt.

Take this data and put the relevant data under these column headers. Return it as a CSV.
school names, school year, address, grade levels

3. The AI returned just the data I needed under those columns, and I uploaded the CSV into one big spreadsheet with every year's data under the same columns.

I also needed to know what type of school each school was, but this data wasn't included in a column. Instead, the schools were sorted by sheet. Public schools were in their own sheet, charter schools in another, and private schools in a third sheet. But some school years also had overlap, so that the same school was listed on the public sheet and the charter sheet (obviously indicating that it was a public charter school). Instead of going through and manually checking where each school was listed, this is how I did it.

I gave this prompt to AI:

Take this data and compare it to the following lists.

[insert list of all the schools in a particular school year]

List 1:
[copy and paste all the schools listed on public school sheet]

List 2:
[copy and past all the schools listed on charter school sheet]

List 3:

[copy and paste all the schools listed on private school sheet]

Create a CSV file with school names in one column, and the type of school in the other column. If the school is repeated in List 1, add public to type column. If the school is repeated in List 2, add charter to type column. If the school is repeated in List 3, add private to type column. Some schools may be on more than one list. Return as a CSV.

It returned a CSV file with the type of school in a column next to every school name, which I inserted into my master spreadsheet. TIP: If an AI model is only returning part of a list of data you gave it, and you know some of it is missing, say something along the lines of:

Your responses are being truncated. Please separate your CSVs into multiple parts.

Making scraper data more accessible

I wrote an article about this for RJI as part of my fellowship, so you can read the full explanation here. But essentially, Stephan at GSN made a scraper to pull data about Entergy power outages from Entergy's website. But Entergy only displays current power outages and we wanted reporters to be able to go back and look at outages over the region at really any point in time. Stephan had this data in spreadsheet form but it wasn't very easy to instantly look at and understand.

So through a lot of steps, I eventually converted that data into a map that displayed the power outage data at intervals of ten minutes. Users can now choose a point in time to see what the power grid looked like, and compare it to other points in time. The finished map is publicly available here.

Heat data sonification project

We got data from the NOAA on the average yearly temperature starting in 1895. Drew wanted to represent the rapid increase in temperature, which started around 1980, using a brass band (see this deep dive into data sonification). The temperature increases are split into four periods, each one associated with a tempo that increases as temperatures rise.

100 BPM	1895–1970s	average temp increase by 0.13°F, stable temp over 75-year period
120 BPM	1980–2000	68.82°F

140 BPM 2001–2012 69.36°F

160 BPM 2013–2024 70.29°F

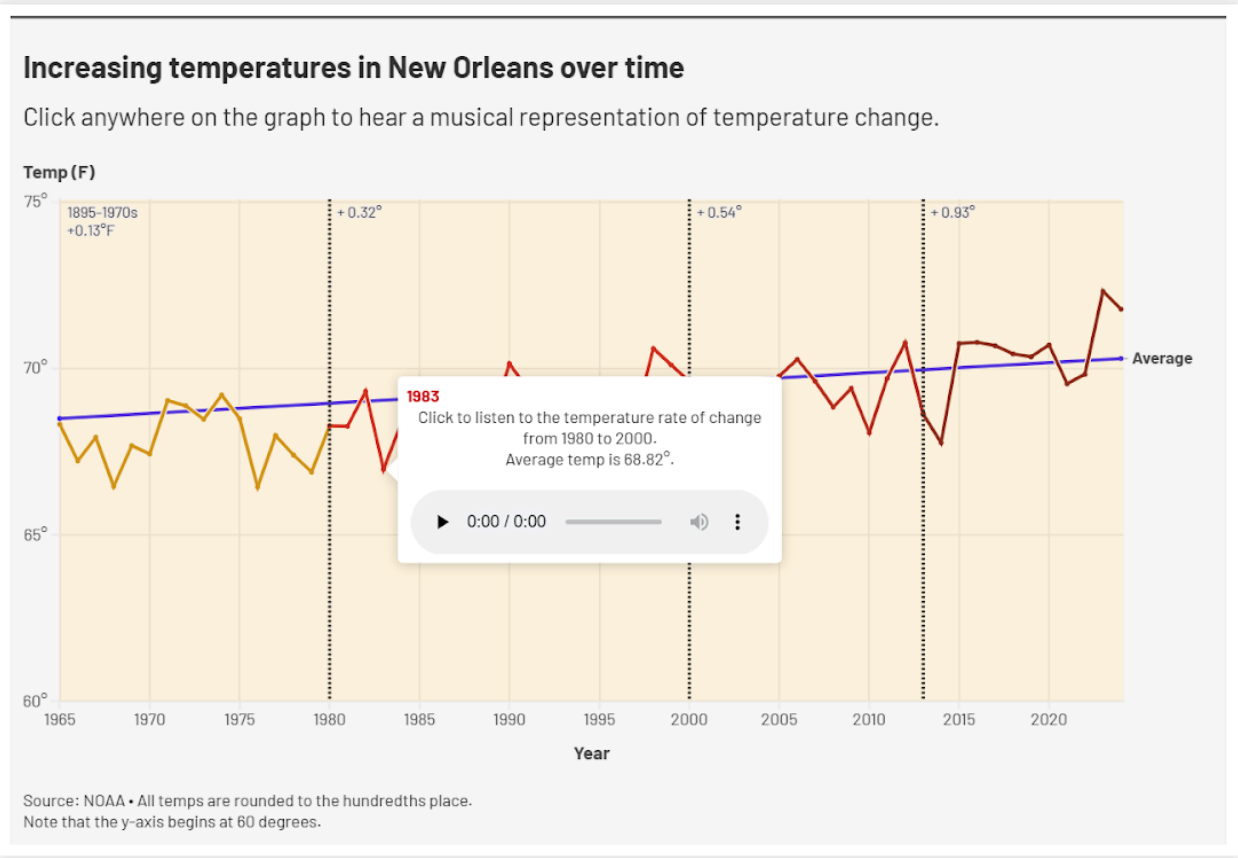
I mapped the yearly temperature data onto a line graph, starting in 1970. Then I added a line to show the average over each interval. We decided to color the lines to gradually get darker red.

Then, I converted the recording of the band playing into an MP3, and split it into four parts (one for each tempo/time period). I embedded them in Flourish using HTML so that you can click on any point on the graph and you can play the music that represents the temperature for that point in time.

This is the HTML, which I put in a column called *Audio.*

```
<p style="text-align:center;">Click to listen to the temperature rate of change from 1895 to 1979. <br> Average temp is 68.5°.</p><br>
<div class='audio-container'>  <audio controls style=margin: 0 auto; display: block;> <source src="https://od.lk/s/MjdfNjI5MTA5MzZf/JoeAvery1.mp3">  Your browser does not support the audio element.  </audio></div>
```

In order to embed an MP3 as a link, so that anyone can listen to it, you have to convert it. I used OpenDrive for this. You just upload the audio file and it'll give you a shareable link to plug into your HTML. Then, under *Popups and Panels,* I put the name of the column containing the HTML, so every data point will have the right audio synced to it.



Index

Boolean data

Data that is classified as true or false values.

String data

Data that is treated as text, whether it is letters, characters or numbers.

LLM (large language model)

This is another word for AI like ChatGPT or Google AI Studio.

HTML (HyperText markup language)

This is the standard coding language for making web pages.

KML (keyhole markup language) file

A file that contains information about geography, including shapes, lines and points. It always contains latitude and longitude but might also contain more information like tilt or altitude if

the shapes are 3D. It's easy to use in GIS software, but for lower-tech data viz projects like Flourish maps, you'll need to convert it first.

API (application programming interface)

An API is how one software shares data with another, without those softwares having to work the same way. An API is essentially the digital middleman that interprets a request for data, retrieves the data, and transmits it back to the requester. For example, a tool like PayPal connects you from your bank to an online shopping site without the bank or the website having to directly communicate. PayPal is an API that protects your financial data from being directly accessed by a third party site.

CSV (comma separate values) file

A file that contains any type of data separated by a delimiter. A delimiter is any symbol that separates units of data, but it is most often a comma. CSVs can be opened as text files, but it's easiest to upload them into a spreadsheet, as each data point will automatically be separated into its own cell. This is what a sample file looks like:

University of Missouri,public,undergrad,graduate,Columbia,Missouri
Truman State,public,undergrad,graduate,Kirksville,Missouri
Stephens College,private,undergrad,graduate,Columbia,Missouri

JSON (JavaScript Object Notation) file

YAML file (Formerly Yet Another Markup Language, currently YAML Ain't Markup Language)

This is a file in the .yml or .yaml format, which is similar to JSON files but has greater capabilities. YML files require correct indentation to run without error (but the indentations are created by spaces, not tabs). They are formatted like this:

```
colleges:
  - name: MU
    city: columbia
  - name: KU
    city: lawrence
  - name: WashU
    city: st. louis
```

This is a simple version, they can be more complicated with more tiers.

GeoJSON (JavaScript Object Notation) file

This is a file containing geographic information that can be read as text by a human or be read by a computer. GeoJSON files can also contain non-spatial information about shapes, lines and points (such as the name of a location, or other information you want to link to your geographic data). The code for a GeoJSON file looks like this:

```
{
  "type": "Feature",
  "geometry": {
    "type": "Point",                    ← this tells the computer the type of shape
    "coordinates": [125.6, 10.1]        ← this is the geographical information
  },
  "properties": {
    "name": "Dinagat Islands"           ← this is the non-geographical information
  }
}
```

Tooltip

This is the name for the information that pops up when you hover over a feature on a website, map, chart or other digital feature. These are sometimes called pop-ups.

Every tool listed in the Toolkit

red = coding | orange = geography | yellow = AI | green = data |

Tool	Function
GitHub	A developer platform for creating and sharing code. There is a free web version and a free desktop app you can download.
JSON validator geoJSON validator Python syntax checker CSV validator HTML validator HTML preview	These are all validators to check that your files are formatted correctly.

XML validator	
Geocodio	Converts locations into coordinates. Can do one at a time or allows you to upload a spreadsheet.
mygeodata, convert to CSV, honeycomb maps, or quickmaptools	These all convert coordinates to addresses and vice versa.
Google Earth	A more advanced version of Google Maps helpful for finding geographic data. There are free web and desktop app versions.
Python	A coding language that is helpful for manipulating data.
Google AI Studio	
Pandas	A tool within Python (must be downloaded to use) that allows you to work with DataFrames.
Compare Sheets extension	A Google Extension within Google Sheets that helps you compare columns, rows and cells to check for duplicate data.
Tabula	A free online tool that converts PDFs into more easily accessible files.
Adobe Acrobat	Software for working with PDFS. It can convert PDFs into other formats. There is a free and pro version.
PDF Miner	A tool for working in Python that extracts text from PDFs.
AIMistrall Small 3.1 AI	a French AI startup with a chatbot and API integration
Grog (not the same as X's grok!)	An AI chatbot
Google AI Studio	Google's AI chat bot, which is more customizable than ChatGPT and other similar software. It's a "thinking" model, which just means it can explain the process behind its response if you want it to. This can help with troubleshooting.
Claude AI	AI with typical chatbot features and has built-in games and workshops for learning skills like coding and languages
Rolli	A search engine to find an expert to interview on most topics.

Google's Data Commons	A platform that hosts lots of data sets. You'll have to verify the accuracy of each data set but it's a good place to start looking.
Harvard's Dataverse	A massive collection of academic/scientific research compiled by Harvard.
New Orleans Open Data	Public data maintained by the city for New Orleans.
French Institute for Demographic Studies	Demographic data maintained by a public research institution in France.
Pew Research Datasets	Datasets collected by Pew Research Center.
Washington Post's climate change data set	Climate change data collected as a CSV on GitHub by the Washington Post.
Louisiana Dept, of Health Eat Safe Inspections	Records of kitchen inspections in Louisiana
Sentencing Project data on people serving life sentences	
Coolors	A tool for creating color palettes
Tencent EdgeOne	Converts MP3 files to shareable links for embedding in HTML.
Tableau Public	
OpenDrive	Converts MP3 files into shareable links.
Data cleaning tips	

Submit feedback on this toolkit here!