

Using Google Colab and Alma's Analytics API to create a Library Data Dashboard

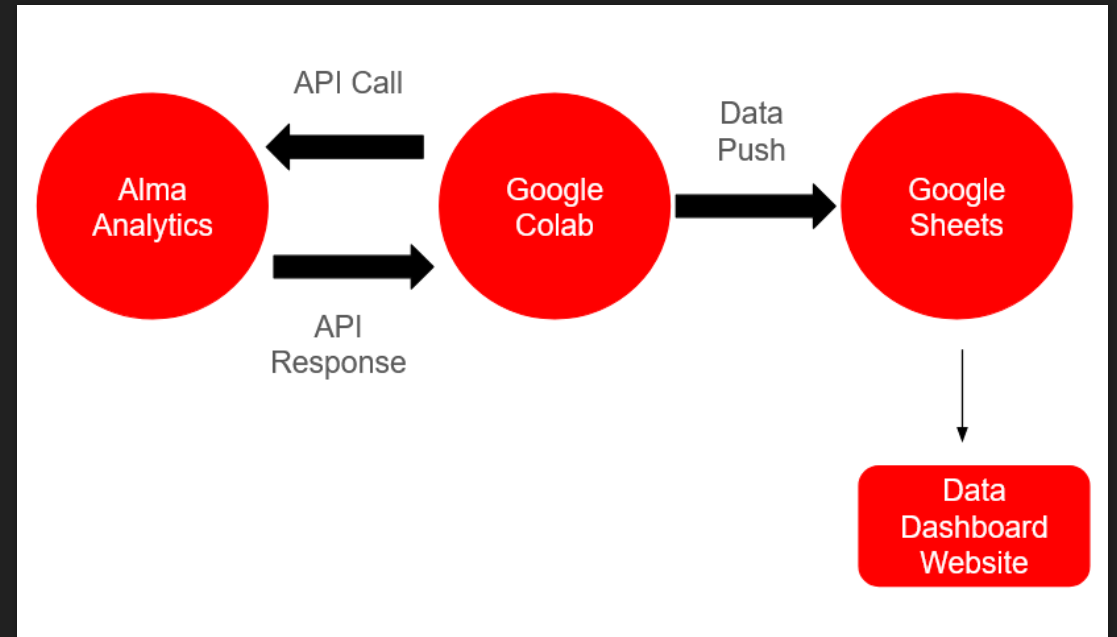
Jill Locascio, MLIS

Librarian - Systems, Digital & Technical Services

SUNY College of Optometry

The Process (at a glance)

- Python script in Colab makes the API call
- Alma sends response to Colab
- Colab parses data and sends to Sheets
- Charts are created in Sheets
- Embed code for charts is placed on website



But first! (obligatory disclaimers)

- This presentation is about getting data from point A to point B
- This presentation is not about best practices for creating visualizations or telling stories with data
- There is code involved but you do not have to be a coder
- I'm a novice at coding - the script works but could certainly be better!

Our data journey

The “Before times”:

No real system for gathering, maintaining, sharing statistics. Data scattered and siloed.

2023 (Data Dashboard):

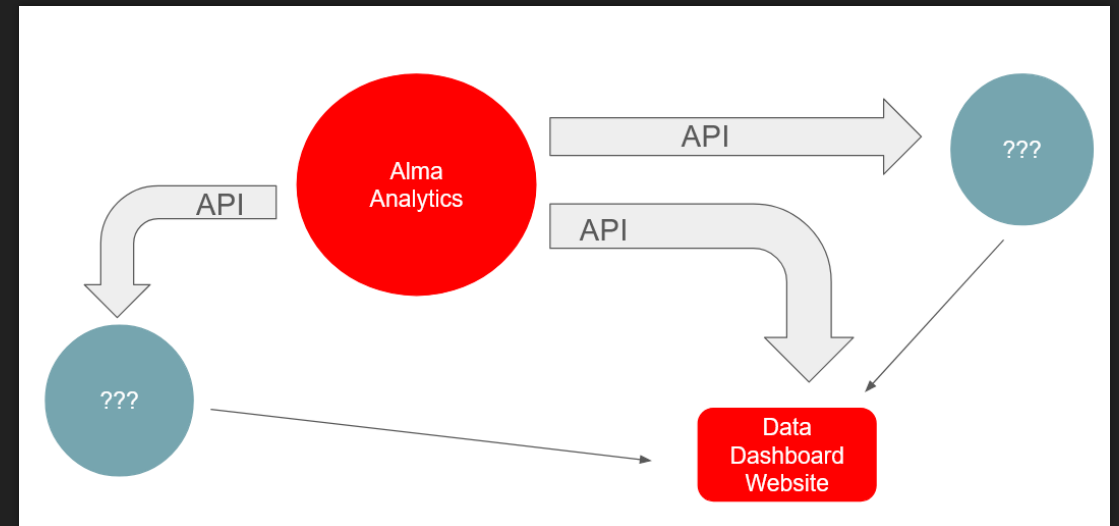
Data taken from Alma Analytics shared in a visually appealing manner on an easy to access website.

2019 (Alma Analytics):

A big improvement in organizing and maintaining data but doesn't address issues with sharing data.

Getting data from Alma Analytics

- There are many methods to get data from Alma Analytics and display it on a webpage
- All of these methods rely on Alma's Analytics API



Google Sheets as an intermediary

- Original tabular data w/o visualizations can be viewed and shared
- Can create attractive visualizations
- Visualizations can be embedded on a website
- Published visualizations update automatically
- Free

Q: Is it possible to import Alma Analytics data into a Google Sheet?

A: Yes!

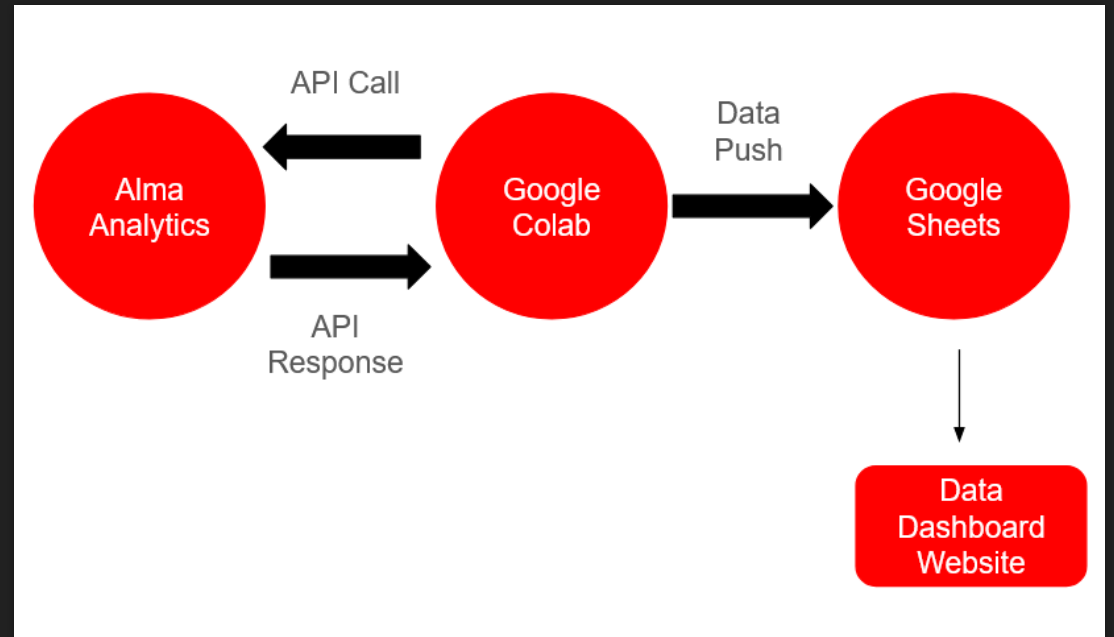
What is Google Colab?

- Python development environment that runs in the browser
- No need to install Python on your computer
- All that you need is a Google account

A python script run in Google Colab can pull data from Alma using their Analytics API and push that data into a Google Sheet.

The Process (at a glance - again)

- Colab makes API call
- Alma sends response to Colab
- Colab parses data and sends to Sheets
- Charts are created in Sheets
- Embed code for charts is placed on website



Set-up Overview

1. Google Sheets

- a) Create main Google sheet
- b) Create worksheets within main sheet for each report

2. Alma

- a) Log-in to Developer Network
- b) Create Alma Analytics API key (if you do not already have one)
- c) Go to (or create) Analytics report(s) (reports for dashboard must be in shared folder)
- d) Copy the path to your report(s)
- e) Test API call(s) in Developer Network console
- f) Use Request URL(s) to examine returned XML

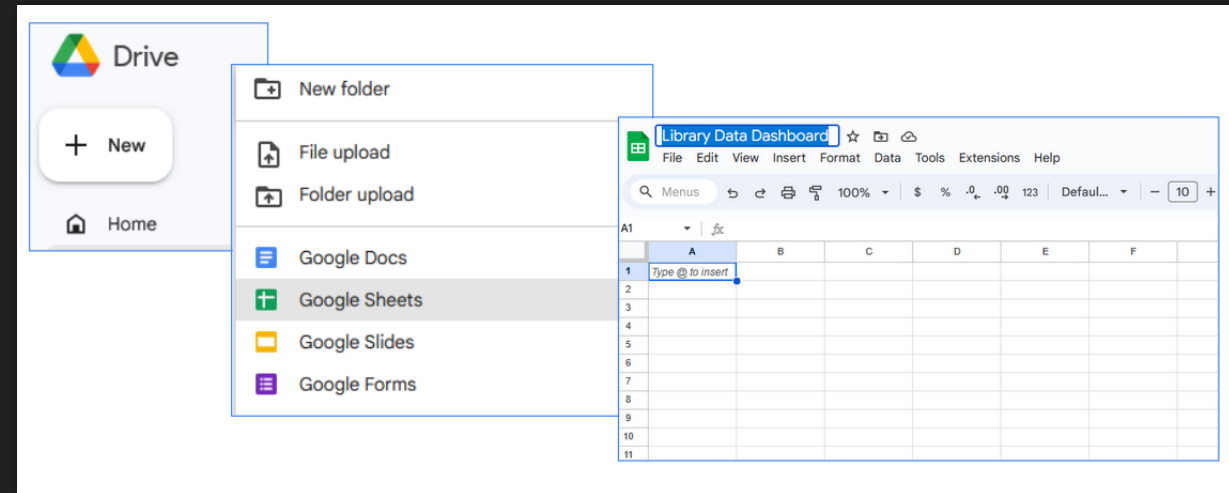
3. Google Colab

- a) Open new Colab notebook
- b) Paste code from template
- c) Modify code so it reflects your API Key, Report Path(s), Column Mapping(s), etc.

Google Sheets

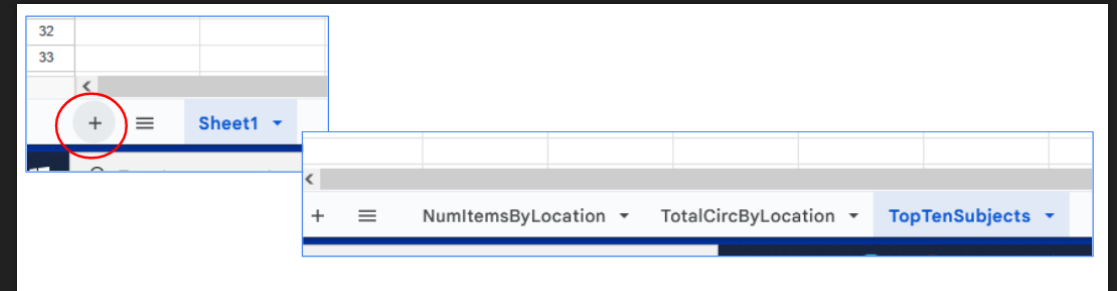
Open and name a new Google Sheet

- This will be where the imported data lives
- You will create your visualizations from the data that gets imported here



Create and name individual worksheets


- 1 worksheet for each report



Alma

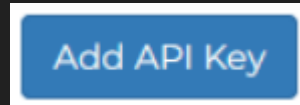
Log-in to Ex Libris Developer network

- Developer Network: <https://developers.exlibrisgroup.com/>
- Once logged in go to Build > API Keys - Manage Keys
- Confirm that you have an Analytics API key

Enabled	Name	API Key	Supported APIs	Action
<input checked="" type="checkbox"/>	Analytics for Google Sheets	<div>.....</div>  Copy	<div>• Analytics - Production Read-only</div>	Edit Delete

Create an Analytics API key

- Click Add API Key
- Fill out the form and click Add Permission
- Under Area select Analytics
- Under Env select Production
- Under Permissions select Read-only
- Save



Add/Edit API Key

Name: Analytics API Key

Description: Used in Google Colab for Data Dashboard

Enabled: ☒

Save detailed log files: ☐

Allowed IP Range: Optional - Only requests within the IP Range will be allowed. CIDR format (Example: 1.2.3.4/32)

Restriction Profile code: Optional - the code of the Integration Profile which limits access to specific vendors/libraries.

Permissions

Area	Env	Permissions	Action
Analytics	Production	<input checked="" type="radio"/> Read-only <input type="radio"/> Read/write	Delete

Add Permission

Save Cancel

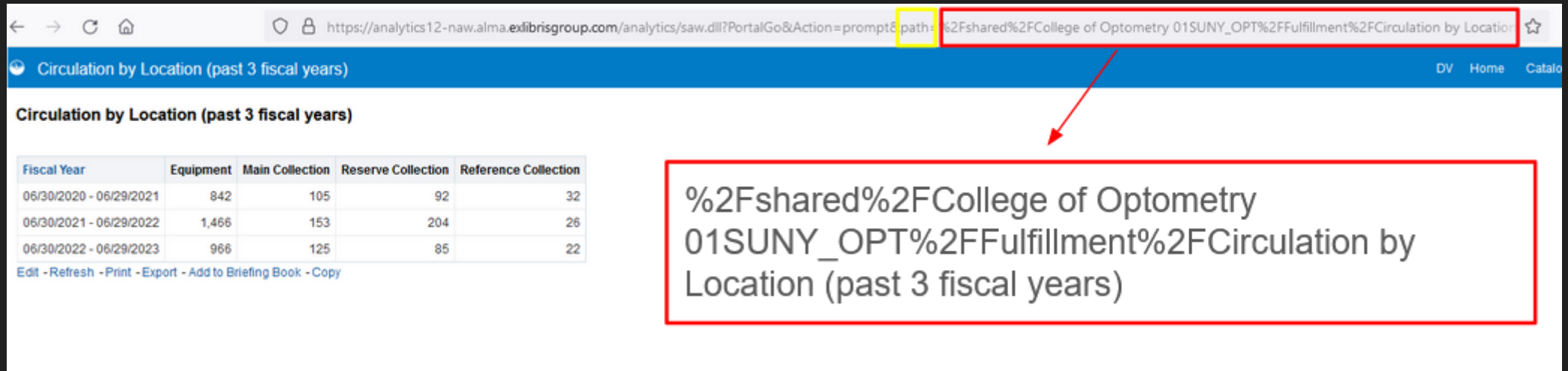
Open Analytics Report(s)

- In Analytics find your report and click Open
- Report must be saved somewhere in your Shared Folders

The screenshot displays the Analytics interface. On the left, a 'Folders' sidebar shows a tree structure: 'My Folders' and 'Shared Folders' (highlighted with a red box). Under 'Shared Folders', there is a folder named 'College of Optometry 01SUNY_OPT' which contains a sub-folder 'Fulfillment' (also highlighted with a red box). The main area shows a report titled 'Circulation by Location (past 3 fiscal years)' with a location path '/Shared Folders/College of Optometry 01SUNY_OPT/Fulfillment' (highlighted with a red box). Below the title, there is a warning 'Used in Data Dashboard. DO NOT DELETE.' and an 'Open' button (highlighted with a red box) next to 'Edit' and 'More' options. The report details include 'Last Modified 1/2/2024 5:59:45 PM' and 'Owner HE_11100_48'.

Copy the path to your Analytics report(s)

- In the address bar of your web browser find where it says “path=” in the URL and copy everything after the equals sign



The screenshot shows a web browser with the URL `https://analytics12-naw.alma.exlibrisgroup.com/analytics/saw.dll?PortalGo&Action=prompt&path=%2Fshared%2FCollege of Optometry 01SUNY_OPT%2FFulfillment%2FCirculation by Location`. The path portion is highlighted in the address bar. A red box and arrow point to the path text, which is also shown in a separate box.

Circulation by Location (past 3 fiscal years)

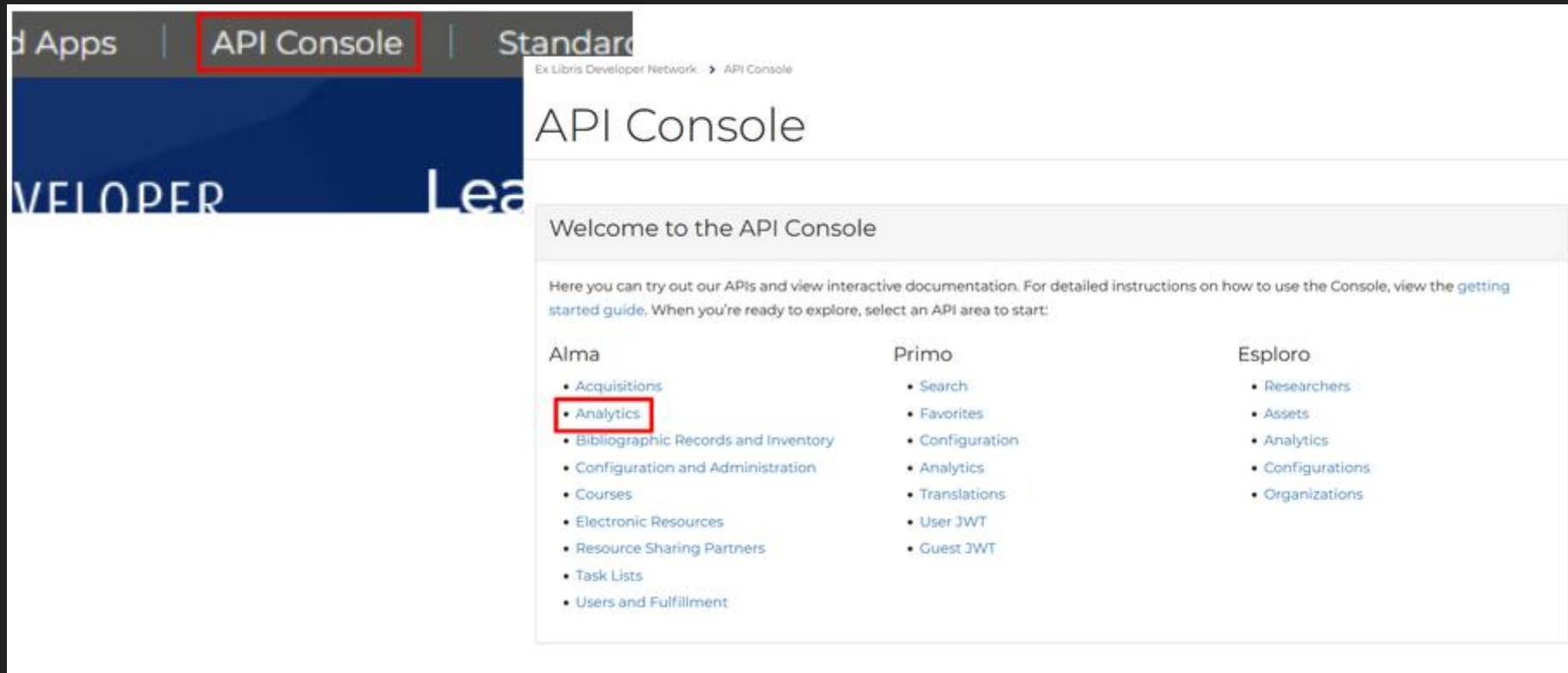
Fiscal Year	Equipment	Main Collection	Reserve Collection	Reference Collection
06/30/2020 - 06/29/2021	842	105	92	32
06/30/2021 - 06/29/2022	1,466	153	204	26
06/30/2022 - 06/29/2023	966	125	85	22

Edit - Refresh - Print - Export - Add to Briefing Book - Copy

`%2Fshared%2FCollege of Optometry
01SUNY_OPT%2FFulfillment%2FCirculation by
Location (past 3 fiscal years)`

API Console

- Return to the Developer Network and go to API Console > Analytics



Prepare your API call

- Select your server and API Key, under Reports click Get
- Click Try It Out and paste your path in the path field

Parameters

Try it out

Name	Description
path string (query)	Full path to the report (URL encoded, taken from the Analytics UI URL). The API can work on the Alma folder or the institutional folder, but not on the community folder. Default value : path - Full path to the report (URL enco

Analytics

Servers

https://api-na.hosted.exlibrisgroup.com

API Key

18xxf78b8d... - Analytics for Google Sheets - Read-only

Reports

GET /almaws/v1/analytics/reports Retrieve Analytics report

Edit Path and execute API call

- Delete encoding characters from path and replace them with forward slashes:

~~%2Fshared%2FCollege of Optometry 01SUNY_OPT%2FFulfillment%2FCirculation by Location (past 3 fiscal years)~~

/shared/College of Optometry 01SUNY_OPT/Fulfillment/Circulation by Location (past 3 fiscal years)

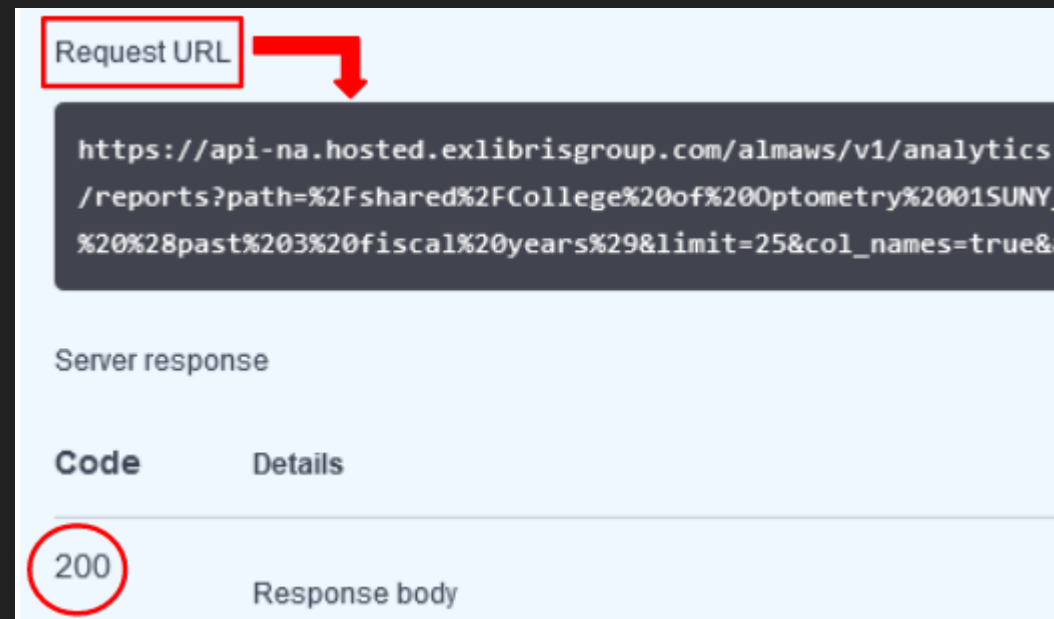
- You do not need to fill out any other fields
- Click Execute

path string (query)	Full path to the report (URL encoded, taken from the Analysis page, not on the community folder).
<input type="text" value="/shared/College of Optometry 01SUNY_C"/>	

Execute

Confirm a successful API response

- Confirm that the call was successful (response code 200)
- Copy request URL and paste it into a new browser tab



The screenshot shows an API client interface. At the top, the 'Request URL' field is highlighted with a red box, and a red arrow points to the URL text area below it. The URL is a long HTTPS string. Below the URL, the 'Server response' section is visible. It contains a table with two columns: 'Code' and 'Details'. The 'Code' column has a value of '200', which is circled in red. The 'Details' column has a value of 'Response body'.

```
Request URL
```

```
https://api-na.hosted.exlibrisgroup.com/almaws/v1/analytics  
/reports?path=%2Fshared%2FCollege%20of%20Optometry%201SUNY_  
%20%28past%203%20fiscal%20years%29&limit=25&col_names=true&
```

Server response

Code	Details
200	Response body

Troubleshooting the API call

- Have you logged in to the developer network?
- Have you selected the correct API key?
- Is your report in your shared folder?
- Is your path correct?
- Have you replaced encoding characters in the path?
- More info: <https://developers.exlibrisgroup.com/alma/>

Examine the API response

- Request URL contains path & API key:

[https://api-na.hosted.exlibrisgroup.com/almaws/v1/analytics/reports?path=\[your path will be here\]&limit=25&col_names=true&apikey=\[your API key will be here\]](https://api-na.hosted.exlibrisgroup.com/almaws/v1/analytics/reports?path=[your path will be here]&limit=25&col_names=true&apikey=[your API key will be here])

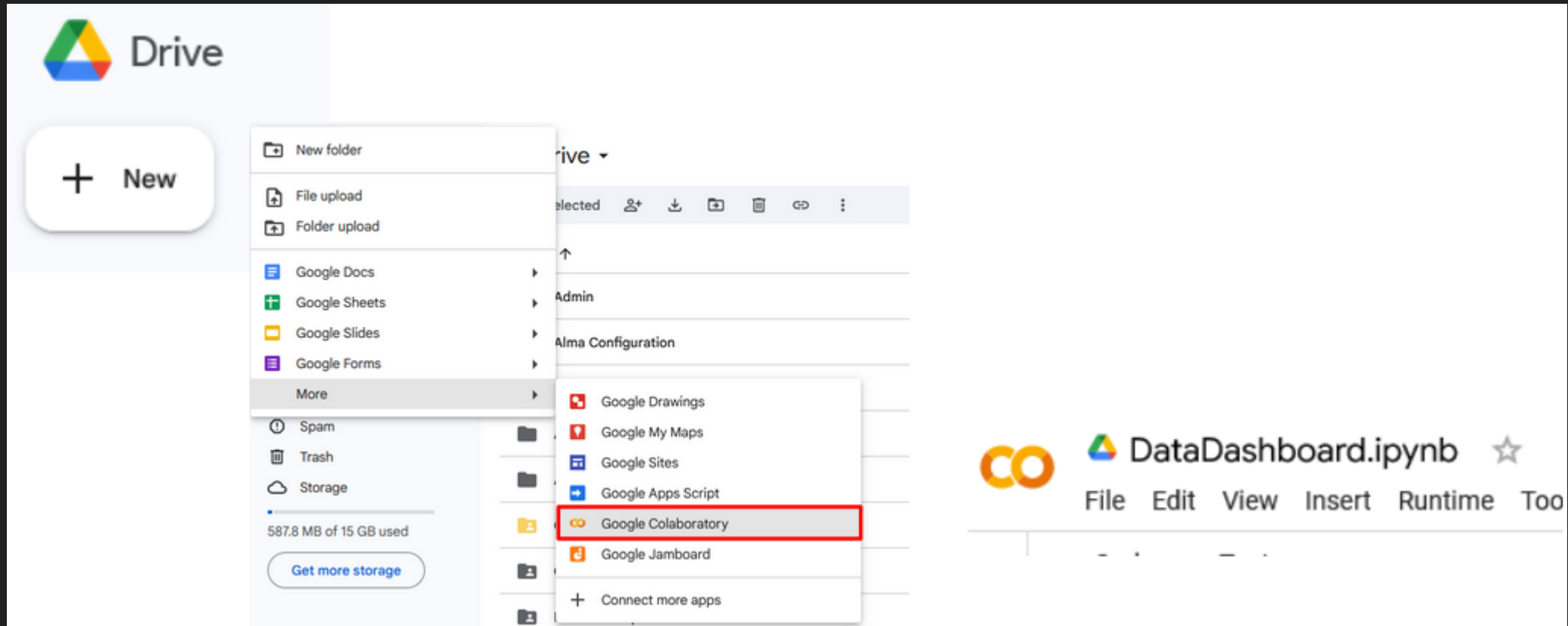
- Copy/paste request URL in new browser tab to view API response

```
sql:precision="255" saw-sql:scale="0" saw-sql:tableHeading="Location" saw-sql:type="varchar" type="xsd:string"/>
<xsd:element maxOccurs="1" minOccurs="0" name="Column2" saw-sql:aggregationRule="sum" saw-sql:aggregationType="agg" saw-sql:column
Repository)" saw-sql:length="8" saw-sql:precision="20" saw-sql:scale="0" saw-sql:tableHeading="Physical Item Details" saw-sql:type="double" t
</xsd:sequence>
</xsd:complexType>
</xsd:schema>
<Row>
  <Column0>0</Column0>
  <Column1>Main collection</Column1>
  <Column2>19868</Column2>
</Row>
<Row>
  <Column0>0</Column0>
  <Column1>Periodical Collection</Column1>
  <Column2>9085</Column2>
```

Google Colab

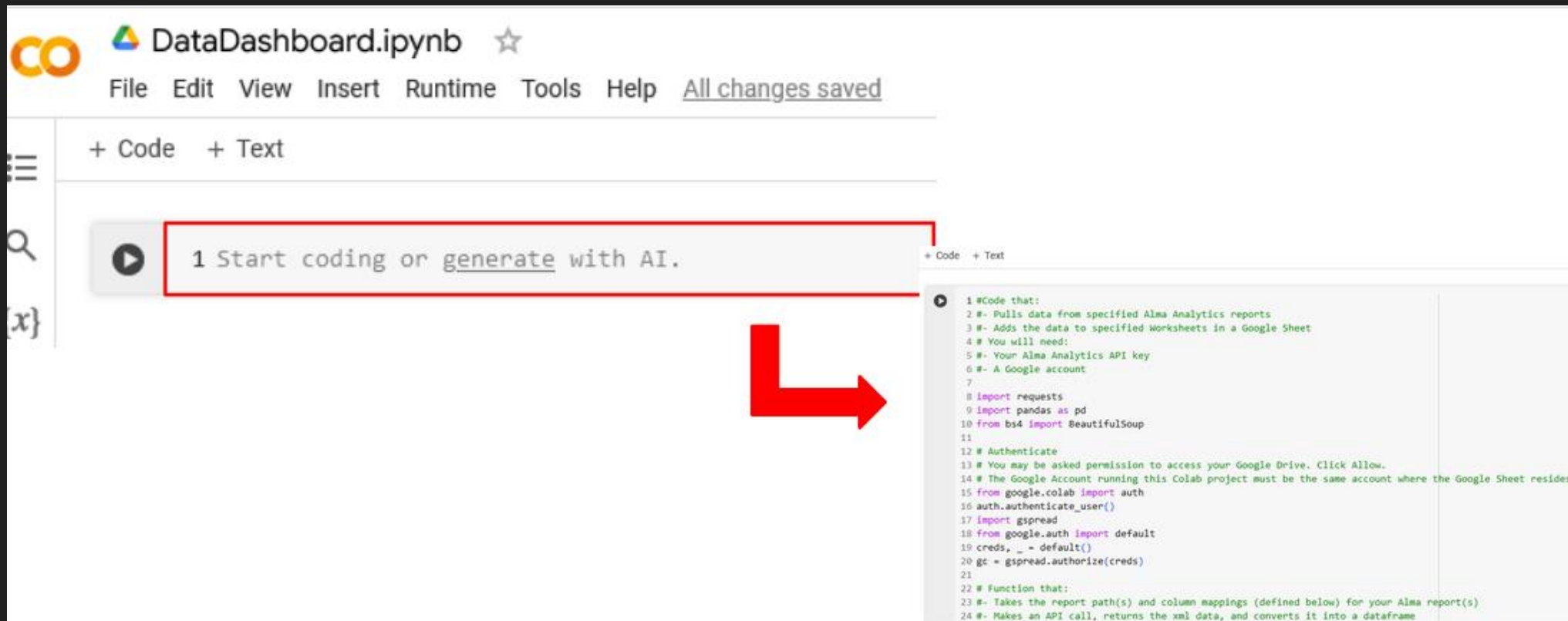
Open and name a new Colab notebook

- This is where you will be pasting the code from the template



Copy/paste the code from the template

- Template code: https://colab.research.google.com/drive/1VjRUa04u-PBWFZzf11vSYMI53h_6NI28?usp=sharing



The screenshot displays a Google Colab notebook interface. At the top, the notebook is titled "DataDashboard.ipynb". Below the title bar, there is a menu with options: File, Edit, View, Insert, Runtime, Tools, and Help. A status message "All changes saved" is visible. The main workspace shows a code cell with the text "1 Start coding or generate with AI.". A red rectangular box highlights this cell. A large red arrow points from the highlighted cell to the right, where a code editor is open, displaying the following Python code:

```
1 #Code that:
2 #- Pulls data from specified Alma Analytics reports
3 #- Adds the data to specified Worksheets in a Google Sheet
4 # You will need:
5 #- Your Alma Analytics API key
6 #- A Google account
7
8 import requests
9 import pandas as pd
10 from bs4 import BeautifulSoup
11
12 # Authenticate
13 # You may be asked permission to access your Google Drive. Click Allow.
14 # The Google Account running this Colab project must be the same account where the Google Sheet resides
15 from google.colab import auth
16 auth.authenticate_user()
17 import gspread
18 from google.auth import default
19 creds, _ = default()
20 gc = gspread.authorize(creds)
21
22 # Function that:
23 #- Takes the report path(s) and column mappings (defined below) for your Alma report(s)
24 #- Makes an API call, returns the xml data, and converts it into a dataframe
```

Modify code in Colab notebook (overview)

- Line 29: **API Key**; can be found in Developer Network Console and Request URL
- Lines 69-71: **Column number - name mappings**; can be found in API response
- Line 61: **Names of columns with numerical data**; can be found in your column mappings (lines 69-71)
- Lines 78, 85, 92: **Report paths** (one for each report); can be found in Developer Network Console and Request URL (after copying from Analytics)
- Lines 80, 87, 94: The **name of your Google Sheet** (will be the same on all 3 lines) AND the **names of each individual worksheet** (will be different, one for each report)

Line 29

- Replace the text between the single quotes with your API key

```
28     # Enter your unique API key
29     api_key = 'YOURAPIKEY'
30     # Make the API call
```

- You can find your API key in the Developer Network as well as at the end of the Request URL from earlier console test

```
ars)&limit=25&col_names=true&apikey=l8xxf78b8df8
```

Lines 69-71

- Define column mappings based on the API response for each report

```
<Row>  
  <Column0>0</Column0>  
  <Column1>Main collection</Column1>  
  <Column2>19868</Column2>  
</Row>
```

```
66 # Define column mappings for each API response (i.e., each report being used)  
67 # This will make sure the data is displayed correctly in the Google Sheet  
68 # You may have to look at the XML response first to know which column the data you want is in  
69 column_mappings1 = {"Location": 1, "Number of Items": 2} # In the first report the location na  
70 column_mappings2 = {"Location": 1, "Number of Loans": 2} # In the second report the location n  
71 column_mappings3 = {"Subject": 3, "Number of Items": 4} # In the third report the location nam  
72
```

Line 61

- List all columns with numerical data to convert to integers

```
58 # The data will be returned as strings. You will need to convert the data of any columns with numerical data to integers.
59 # Convert specified columns to integers
60 for column_name in column_mappings.keys():
61     if column_name in ["Number of Items", "Number of Loans"]: # These are names of columns with numerical data in my reports. Edit or add more as needed.
62         df[column_name] = pd.to_numeric(df[column_name], errors='coerce')
63
```

Lines 78, 85, 92

- Paste the unique path for each report

```
77 # Fetch and update data for Sheet1
78 df1 = fetch_api_data("/shared/.../First report", column_mappings1)
79 if df1 is not None:
80     sheet1 = gc.open("YourSheetName").worksheet('Sheet1')
81     sheet1.clear()
82     sheet1.update([df1.columns.values.tolist()] + df1.fillna(-1).values.tolist())
83
84 # Fetch and update data for Sheet2
85 df2 = fetch_api_data("/shared/.../Second report", column_mappings2)
86 if df2 is not None:
87     sheet2 = gc.open("YourSheetName").worksheet('Sheet2')
88     sheet2.clear()
89     sheet2.update([df2.columns.values.tolist()] + df2.fillna(-1).values.tolist())
90
91 # Fetch and update data for Sheet3
92 df3 = fetch_api_data("/shared/.../Third report", column_mappings3)
93 if df3 is not None:
94     sheet3 = gc.open("YourSheetName").worksheet('Sheet3')
95     sheet3.clear()
96     sheet3.update([df3.columns.values.tolist()] + df3.fillna(-1).values.tolist())
97
```

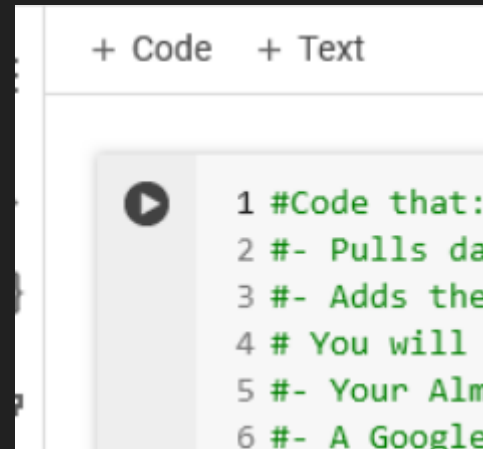

Lines 80, 87, 94

- Paste the name of your Google Sheet and each individual worksheet

```
77 # Fetch and update data for Sheet1
78 df1 = fetch_api_data("/shared/.../First report", column_mappings1)
79 if df1 is not None:
80     sheet1 = gc.open("YourSheetName").worksheet('Sheet1')
81     sheet1.clear()
82     sheet1.update([df1.columns.values.tolist()] + df1.fillna(-1).values.tolist())
83
84 # Fetch and update data for Sheet2
85 df2 = fetch_api_data("/shared/.../Second report", column_mappings2)
86 if df2 is not None:
87     sheet2 = gc.open("YourSheetName").worksheet('Sheet2')
88     sheet2.clear()
89     sheet2.update([df2.columns.values.tolist()] + df2.fillna(-1).values.tolist())
90
91 # Fetch and update data for Sheet3
92 df3 = fetch_api_data("/shared/.../Third report", column_mappings3)
93 if df3 is not None:
94     sheet3 = gc.open("YourSheetName").worksheet('Sheet3')
95     sheet3.clear()
96     sheet3.update([df3.columns.values.tolist()] + df3.fillna(-1).values.tolist())
97
```


Ready to run!

- Scroll up to the top of the code and click the Run Cell button (looks like a triangle in a circle)



Authorization

- You will encounter a series of prompts asking you to allow your Colab notebook to access your Google Sheets – click allow

Allow this notebook to access your Google credentials?

This will allow code executed in this notebook to access your Google Drive and Google Cloud data. Review the code in this notebook prior to allowing access.

No thanks

Allow

As your code runs

- The run cell button will change shape (now a square in a circle)
- You'll be able to see which lines of code are being executed



```
7      api_
8      # Ma
9      r =
10     # Cl
11     if
12
13
14
```

Data in Google Sheet

- Once code has completed running, open your Google Sheet and confirm your data is present

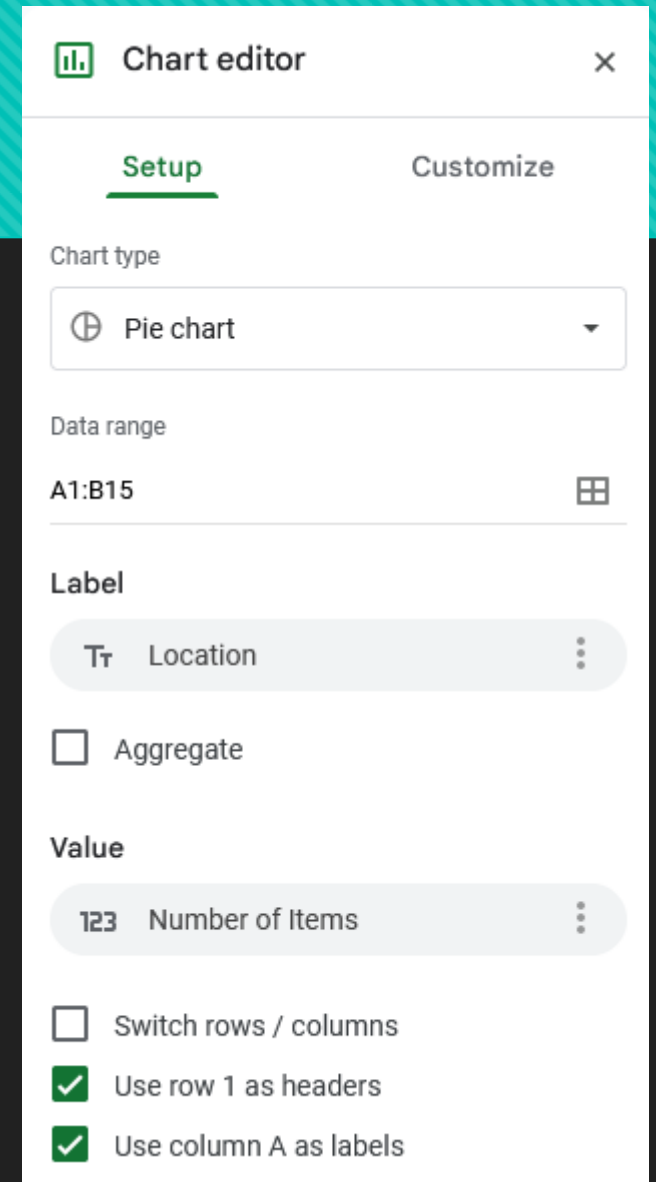
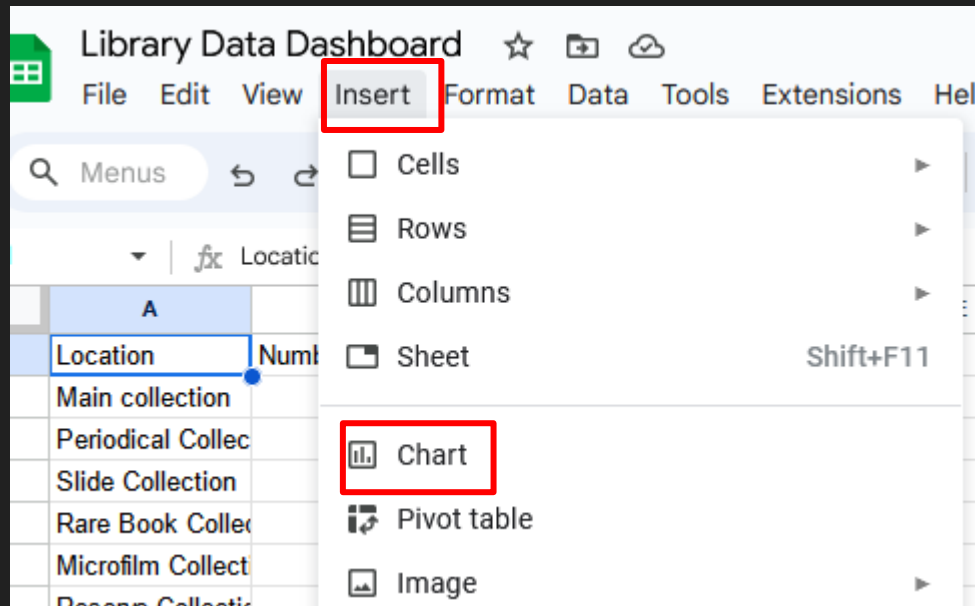
Room Reservations - Column

✓ 33s completed at 3:20 PM

	A	B	C
1	Location	Number of Items	
2	Main collection	19868	
3	Periodical Collec	9085	
4	Slide Collection	3298	
5	Rare Book Collec	304	
6	Microfilm Collect	201	
7	Reserve Collectio	199	
8	Microfiche Collec	191	
9	Video Collection	124	
10	Audio Collection	86	
11	Equipment Collec	85	
12	Reference Collec	58	
13	Pamphlet Collect	19	
14	Study Room	6	
15	CD-ROM Collect	1	

Create your visualizations

- Click Insert > Chart
- Can select from different types of visualizations
- Many options to customize



Publish your visualizations

- Click the three dots at the top of your visualization and select Publish Chart



Select publishing settings

- Go to the Embed tab
- Select “Automatically republish...”
- Publish

Publish to the web

×

This document is not published to the web.

Make your content visible to anyone by publishing it to the web. You can link to or embed your document. [Learn more](#)

Link

Embed

Hourly Circulation ▾

Interactive ▾

Publish

▾ Published content & settings

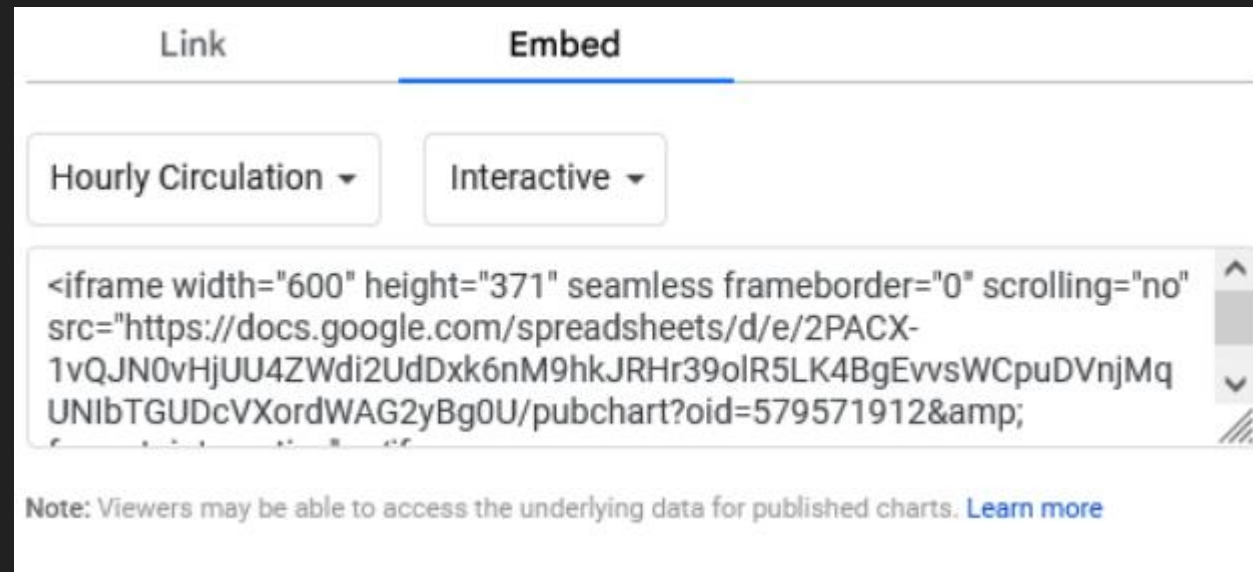
Entire Document ▾

Start publishing

☒ Automatically republish when changes are made

Copy the embed code

- Code consists of <iframe> element to be included in HTML of your web page



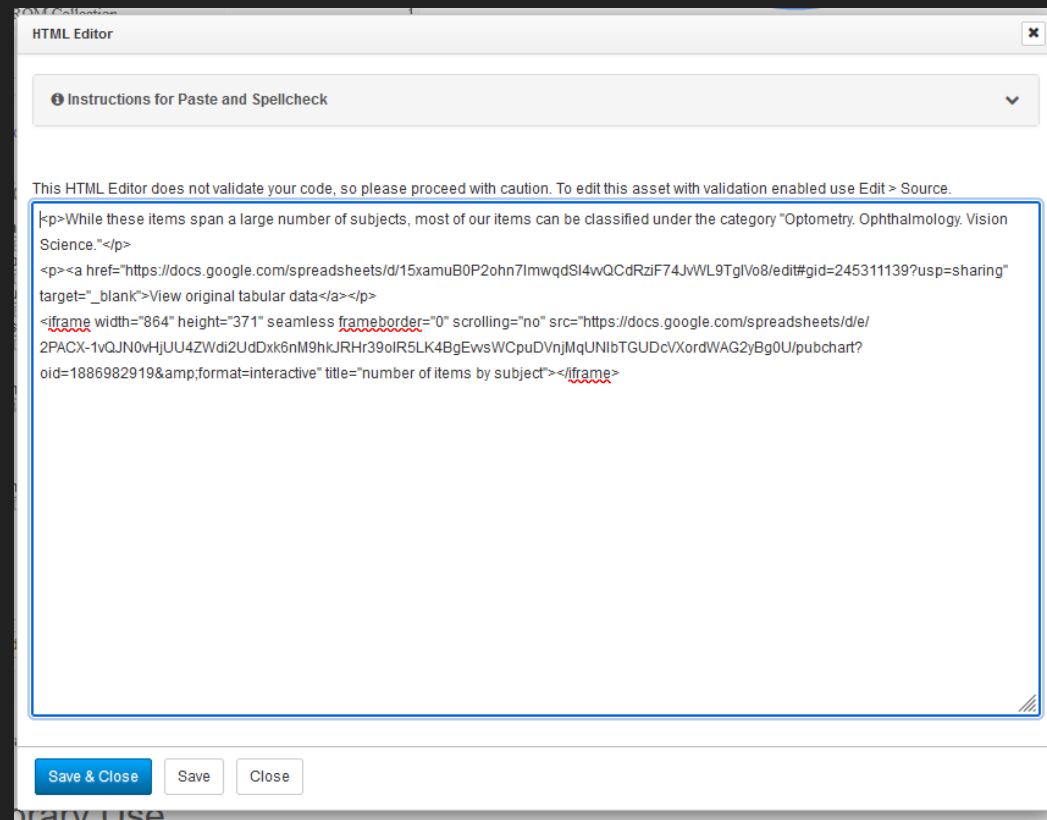
The screenshot shows the 'Embed' tab of a Google Docs interface. At the top, there are two tabs: 'Link' and 'Embed', with 'Embed' being the active tab. Below the tabs, there are two dropdown menus: 'Hourly Circulation' and 'Interactive'. The main area displays the HTML code for an iframe that embeds a Google Sheet chart. The code is as follows:

```
<iframe width="600" height="371" seamless frameborder="0" scrolling="no" src="https://docs.google.com/spreadsheets/d/e/2PACX-1vQJN0vHjUU4ZWdi2UdDxk6nM9hkJRHr39olR5LK4BgEvvsWCpuDVnjMqUNibTGUDcVXordWAG2yBg0U/pubchart?oid=579571912&";
```

Below the code, there is a note: "Note: Viewers may be able to access the underlying data for published charts. [Learn more](#)".

Paste embed code

- Embed code can be pasted in HTML of your data dashboard's web page



All done!

○ Library Data Dashboard: <https://sunyopt.libguides.com/datadashboard>

OUR COLLECTION

We have over 30,000 physical items in our collection.

[View original tabular data](#)

Total Items in the Library

33519

The locations with the most items are our Main Collection, Per

[View original tabular data](#)

Location	Number of Items
Main collection	19868
Periodical Collection	9085
Slide Collection	3298
Rare Book Collection	304
Microfilm Collection	201
Reserve Collection	199
Microfiche Collection	191
Video Collection	124
Audio Collection	86
Equipment Collection	85
Reference Collection	58
Pamphlet Collection	19
Study Room	6
CD-ROM Collection	1

Number of Items

Rare Book Collection

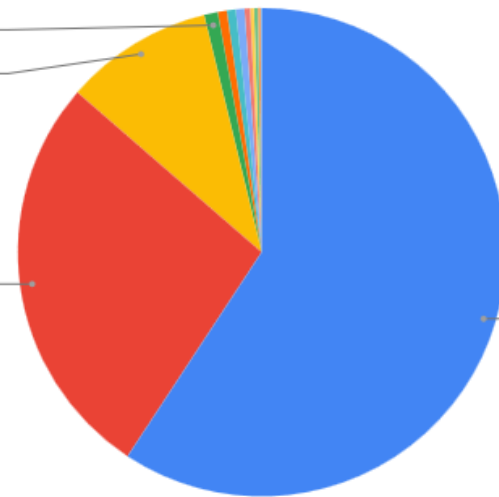
0.9%

Slide Collection

9.8%

Periodical Collection

27.1%



While these items span a large number of subjects, most of our items can be classified under the category "Optometry. Ophthalmology. Vision Science."

[View original tabular data](#)

Top Ten Subjects by Item Count

Theory and practice of education

3.8%

Social pathology. Social and public welfare....

2.0%

Public aspects of medicine

2.6%

Psychology

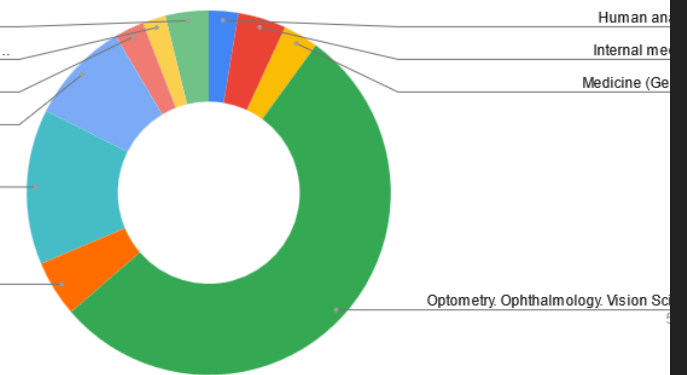
9.0%

Physiology

13.8%

Physics

5.0%



Troubleshooting the process

- Has the location or name of the Analytics report(s) changed?
- Has someone edited the report(s) in a way that may affect the data?
- Have you correctly entered your information in Colab?
- Is the report path correct?
- Are your column mappings correct?
- Has numerical data been converted to integers?
- Is the chart published correctly?
- Are any of your API calls working?

General rule: Retrace your steps, don't be afraid to start from the beginning if needed

Caveats

- Analytics Lag
- Accessibility concerns
- Must run Colab code manually*

* Alternate, non-manual, non-Colab method available here: <https://github.com/gwu-libraries/alma-analytics-sheets>

What's next?

- Incorporate other sources of data
- Broaden the types of data
- Expand scope of dashboard
- Refine visualizations

Questions?

- Email: jlocascio@sunyopt.edu
- Thank you!