

Basic
USAspending API
Training



Agenda

Introduction to APIs

State profile GET request

Advanced Search POST Request

Additional challenges and resources

Questions

Introduction to APIs

When to use the API?

What is an API?

GET vs POST requests

When to use the API?

The USAspending API powers all functionality on the website. Anything you can do on the site, you can do in the API.

The API has some functionality not available on the site.

For many simple and one-off tasks it's often easier to use the website.

- Consider using the USAspending API if...
 - You need functionality which is only available through the API.
 - You want to automate a report you need to run periodically.
 - You want to automate repetitive tasks which would otherwise require manual work on the website.
 - You want to build a workflow that allows you to do more of your tasks in tools like Excel.

What is an API?



What is an API?

The backend team transforms the raw ingredients (data) into more simple and digestible formats.

The frontend team designs beautiful ways to present the data to help humans consume it and gain insights.

APIs bridge this gap by providing the data curated by the backend team in a standard format for presentation on the website, or other tools.

What is an API?

USAspending uses REST API endpoints to transfer formatted data from the server to client browsers.

A REST API endpoint uses a set of defined rules to share or access this formatted data through an HTTP request.

USAspending endpoints each present different data elements with different levels of aggregation and enable different sets of filters.

For example, the set of endpoints used to power state profile pages is different from the set of endpoints used to power advanced search.

Notes: <https://stackoverflow.com/a/18768849>

GET vs POST Requests

- GET and POST are two different types of REST API requests.
- Certain endpoints require you to use either a GET or POST request.
 - This is included in the documentation for each endpoint.
- GET requests are used to get data on a specific record with a known numerical identifier.
- GET requests also support simple filtering.
 - A count of new awards for an agency in a single fiscal year
- POST requests are used in USAspending to support more advanced filtering.
 - All grants that were awarded to a specific congressional district in a specific fiscal year from a specific agency

State profile GET Requests

How much money went to my state in a FY?

Steps:

1. Navigate to a state profile page on USAspending to see the total award amount in a period.
2. Inspect the page to identify which endpoint is used to display this number.
3. Review the endpoint documentation.
4. Adjust the GET request URL for a different state or time period.
5. Replicate this GET request in Excel.

NOTE: The images and values in the following screenshots may appear different on your system.

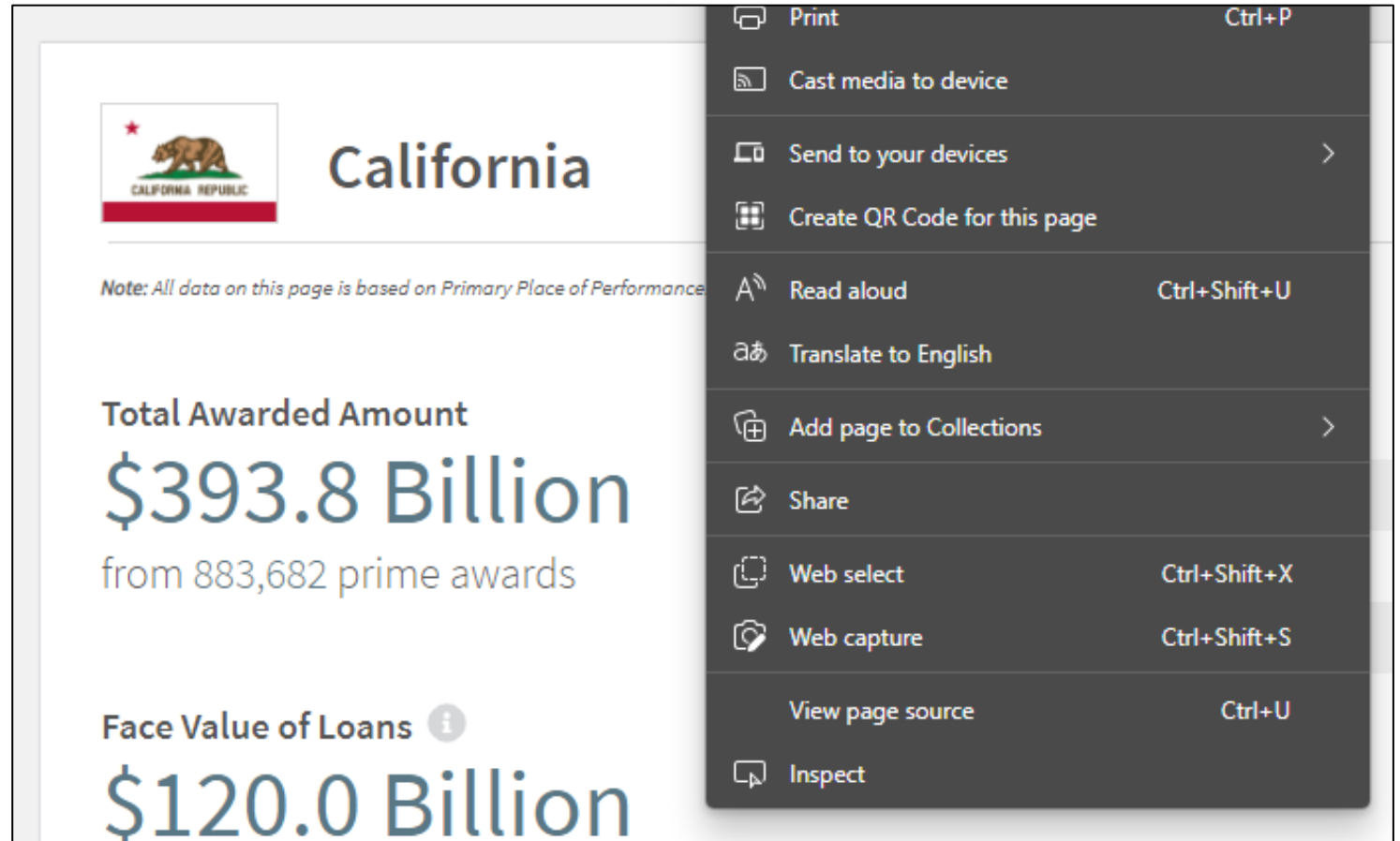
GET Request - Demo

Navigate to the CA state profile page on USAspending:

<https://www.usaspending.gov/state/california/latest>

Observe the Total Award Amount value.

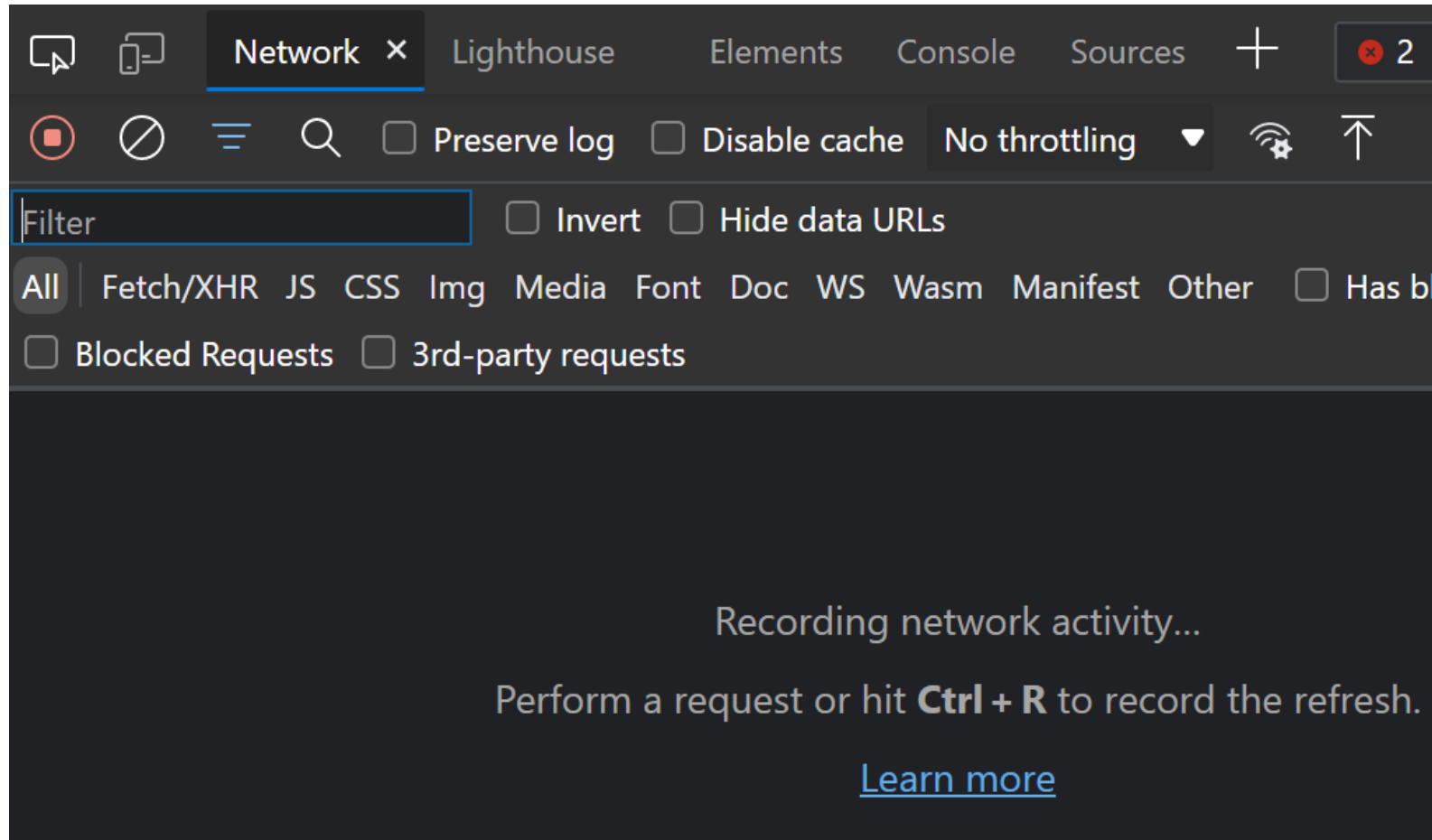
Right click the CA profile page and click Inspect. (or use hotkey Command + Control + c on mac or Control + Shift + c on PC)



The screenshot displays the California state profile page on USAspending. The page features the California state seal and the word "California" in a large font. Below this, a note states: "Note: All data on this page is based on Primary Place of Performance". The main data point is "Total Awarded Amount" which is "\$393.8 Billion" from 883,682 prime awards. Below that, it shows "Face Value of Loans" as "\$120.0 Billion". A right-click context menu is open over the page, listing various actions such as Print, Cast media to device, Send to your devices, Create QR Code for this page, Read aloud, Translate to English, Add page to Collections, Share, Web select, Web capture, View page source, and Inspect. The "Inspect" option is highlighted at the bottom of the menu.

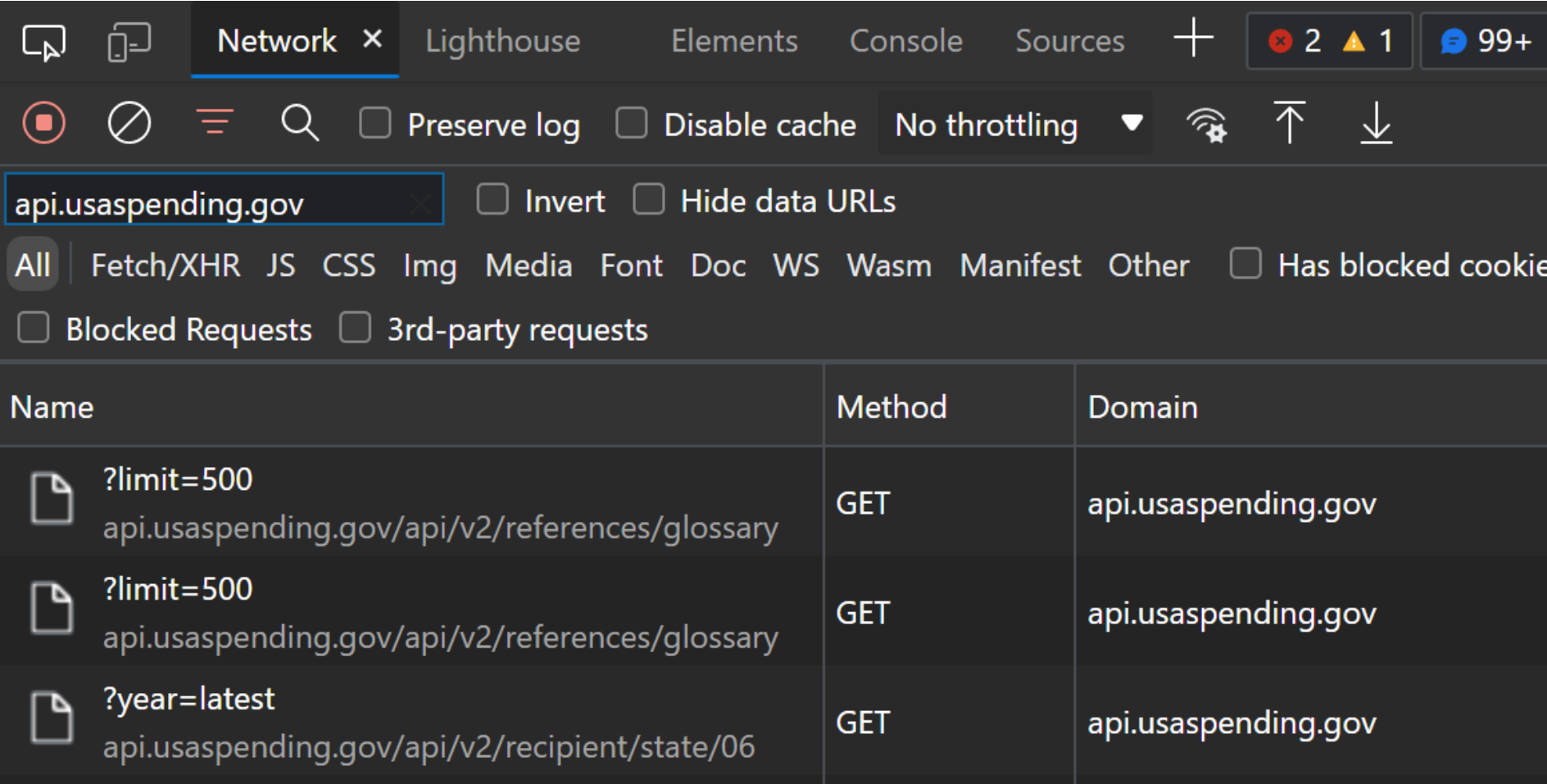
Action	Shortcut
Print	Ctrl+P
Cast media to device	
Send to your devices	
Create QR Code for this page	
Read aloud	Ctrl+Shift+U
Translate to English	
Add page to Collections	
Share	
Web select	Ctrl+Shift+X
Web capture	Ctrl+Shift+S
View page source	Ctrl+U
Inspect	

Demo continued






Select the Network tab and **refresh the page**.

Demo continued



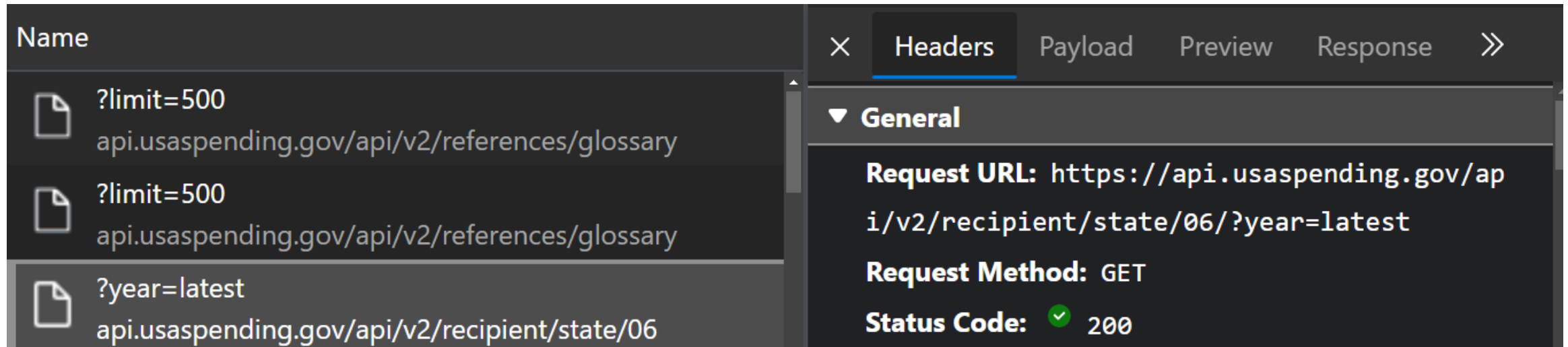
The screenshot shows the Chrome DevTools Network tab with the following settings:

- Filter: `api.usaspending.gov`
- Options: Invert, Hide data URLs
- Request Type: **All** (Fetch/XHR, JS, CSS, Img, Media, Font, Doc, WS, Wasm, Manifest, Other)
- Filters: Has blocked cookie, Blocked Requests, 3rd-party requests

Name	Method	Domain
 <code>?limit=500</code> <code>api.usaspending.gov/api/v2/references/glossary</code>	GET	<code>api.usaspending.gov</code>
 <code>?limit=500</code> <code>api.usaspending.gov/api/v2/references/glossary</code>	GET	<code>api.usaspending.gov</code>
 <code>?year=latest</code> <code>api.usaspending.gov/api/v2/recipient/state/06</code>	GET	<code>api.usaspending.gov</code>

Use the filter box to only show API calls to `api.usaspending.gov`

Demo continued



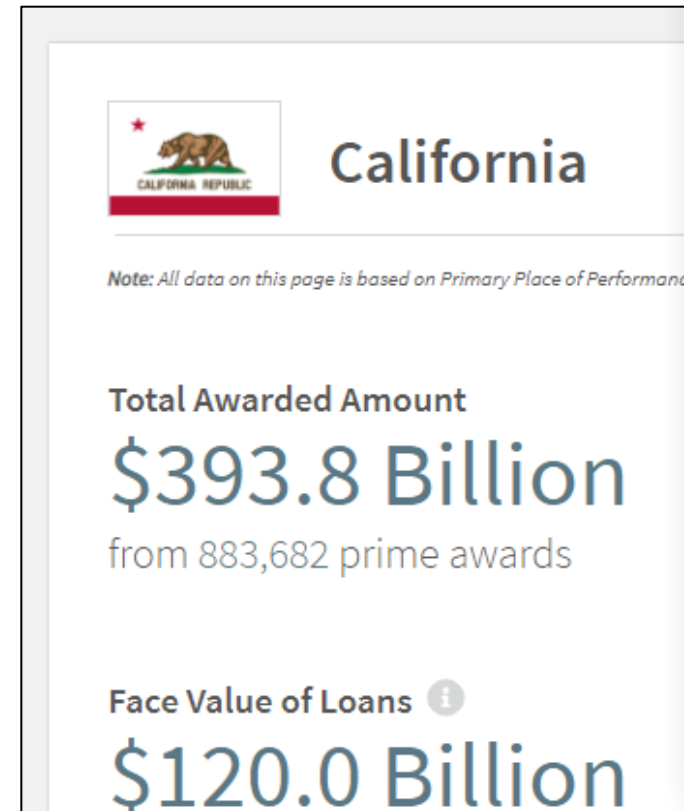
The screenshot displays a web browser's developer tools interface. On the left, a list of network requests is shown under the 'Name' column. The third request, '?year=latest', is selected and highlighted. The right-hand pane shows the details for this request, with the 'Headers' tab active. Under the 'General' section, the following information is visible:

- Request URL:** `https://api.usaspending.gov/api/v2/recipient/state/06/?year=latest`
- Request Method:** GET
- Status Code:** ✔ 200

Select the `api/v2/recipient/state/06` API request.
Observe the Request URL and Request Method.

Demo continued

```
{  
  "name": "California",  
  "code": "CA",  
  "fips": "06",  
  "type": "state",  
  "population": 39536653,  
  "pop_year": 2017,  
  "pop_source": "U.S. Census Bureau, 2017 Populatio  
  "median_household_income": 67739.0,  
  "mhi_year": 2016,  
  "mhi_source": "U.S. Census Bureau, 2016 American  
  "total_prime_amount": 393881512738.49,  
  "total_prime_awards": 884698,  
  "total_face_value_loan_amount": 119983339041.15,  
  "total_face_value_loan_prime_awards": 438551,  
  "award_amount_per_capita": 9962.44  
}
```



Open the Request URL in your browser.

Compare the total_prime_amount and the Total Awarded Amount.

GET Request – Demo

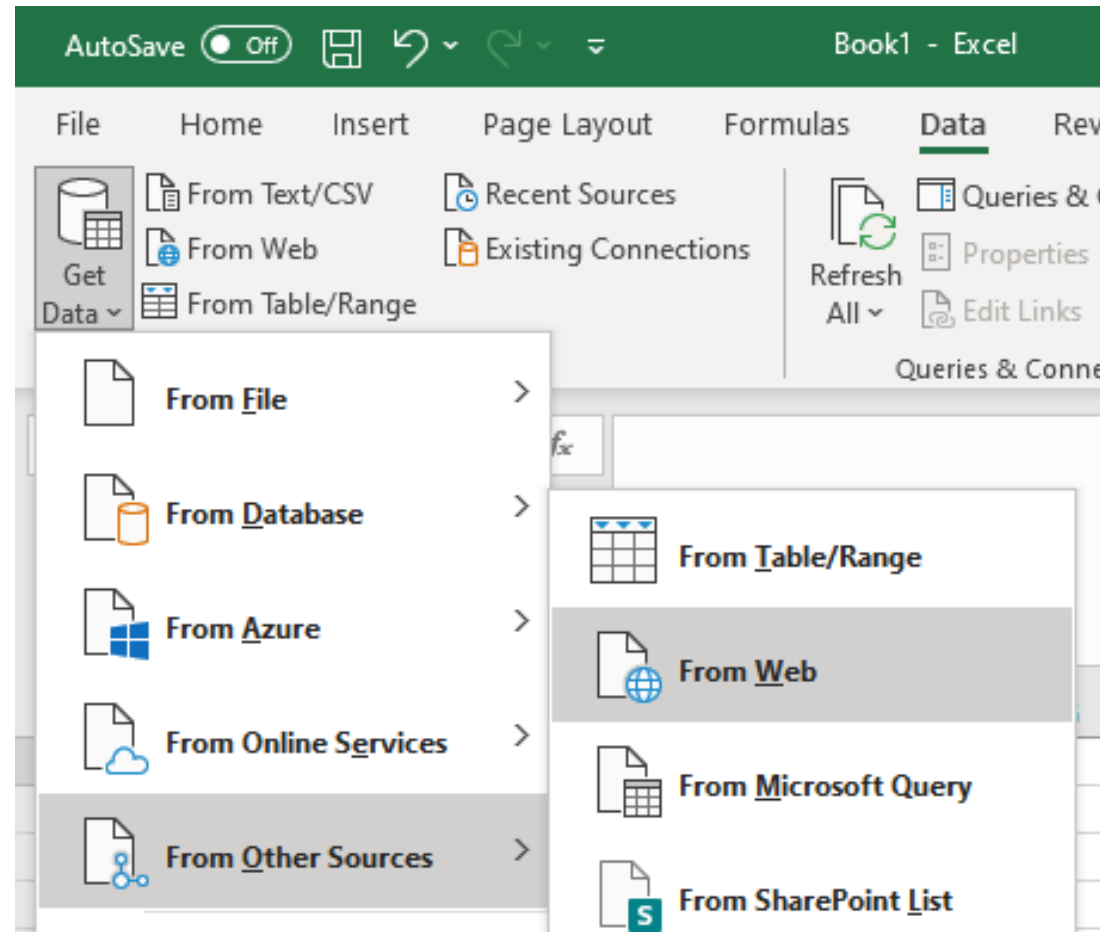
<https://api.usaspending.gov/api/v2/recipient/state/06/?year=latest>

- The “/06/” in the URL specifies a state by FIPS code (CA in this case)
- The “?year=latest” filters the data filters the data to the trailing 12 months

Try changing the “06” to a different FIPS code.

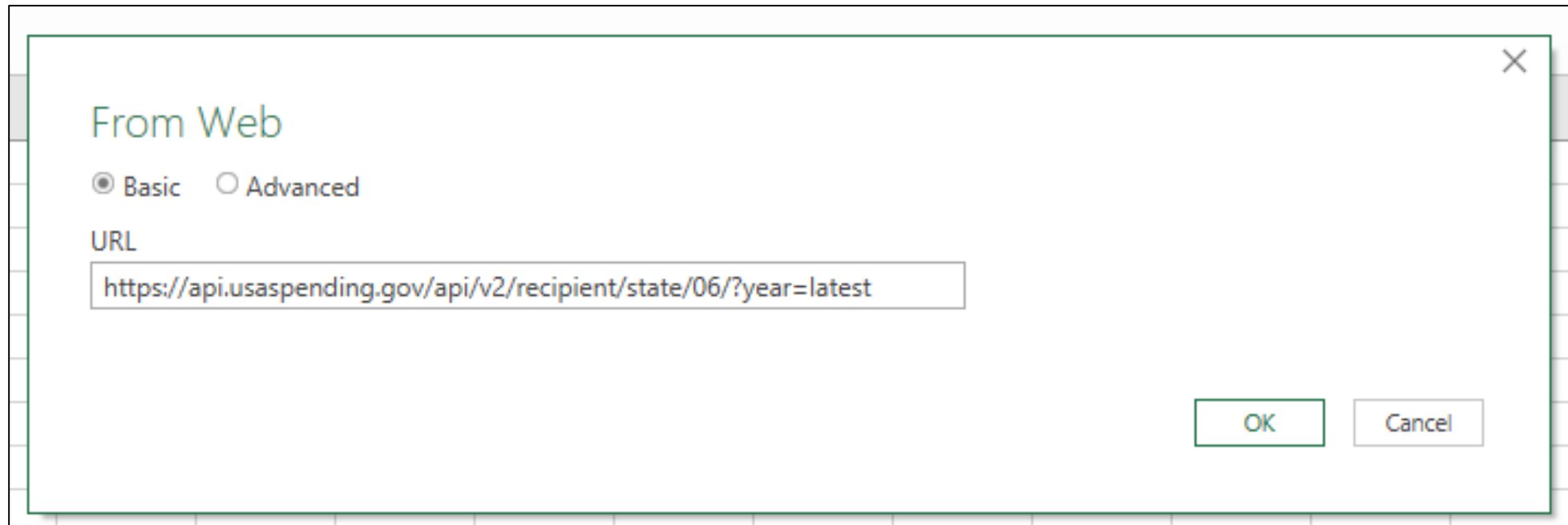
Try changing the year filter to 2022.

Demo continued



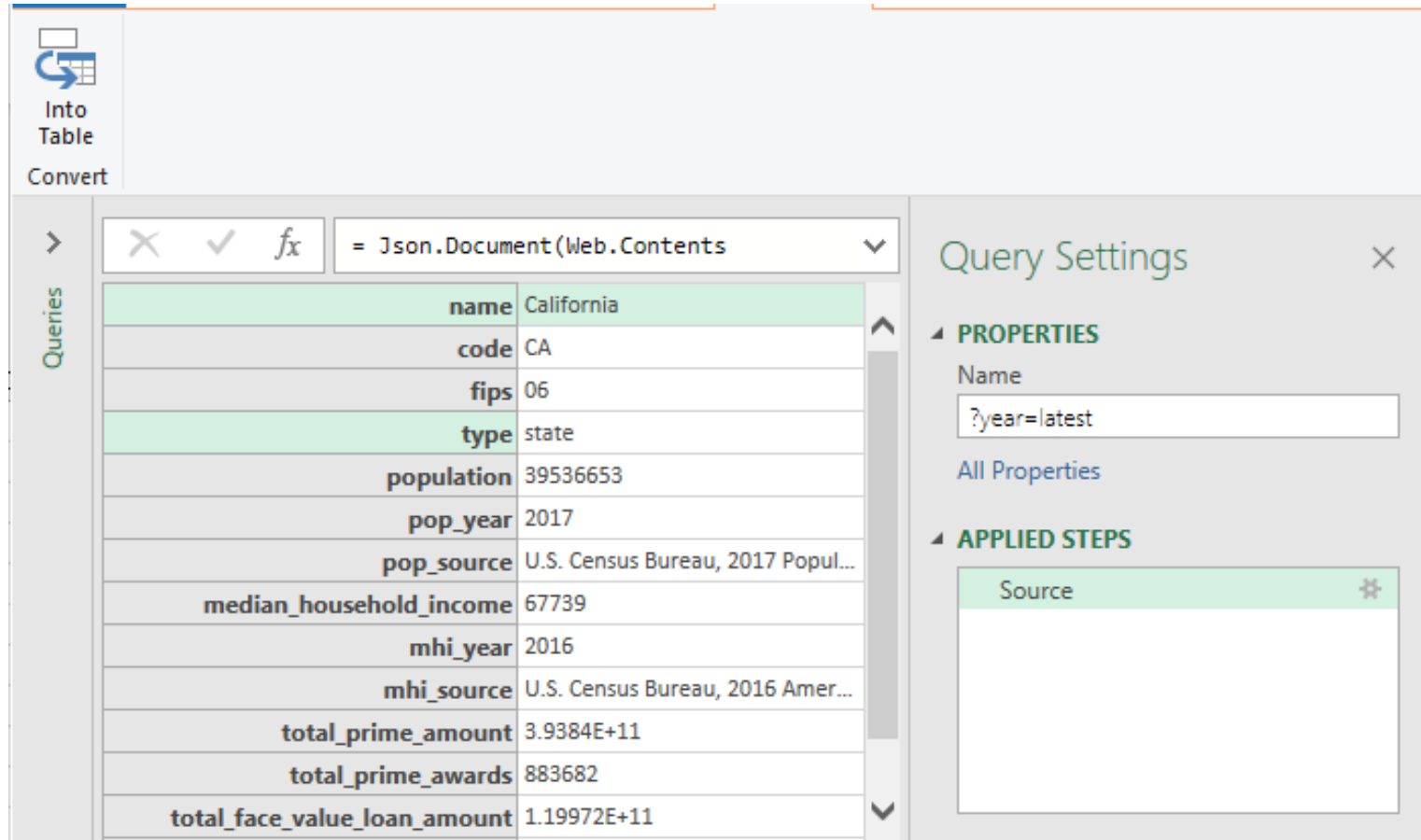
In Excel, select Data > Get Data > From Other Sources > From Web

Demo continued



Paste your GET request URL into the URL box and click OK.

Demo continued



The screenshot shows a web browser interface with a table of data for California and a 'Query Settings' panel. The table has the following data:

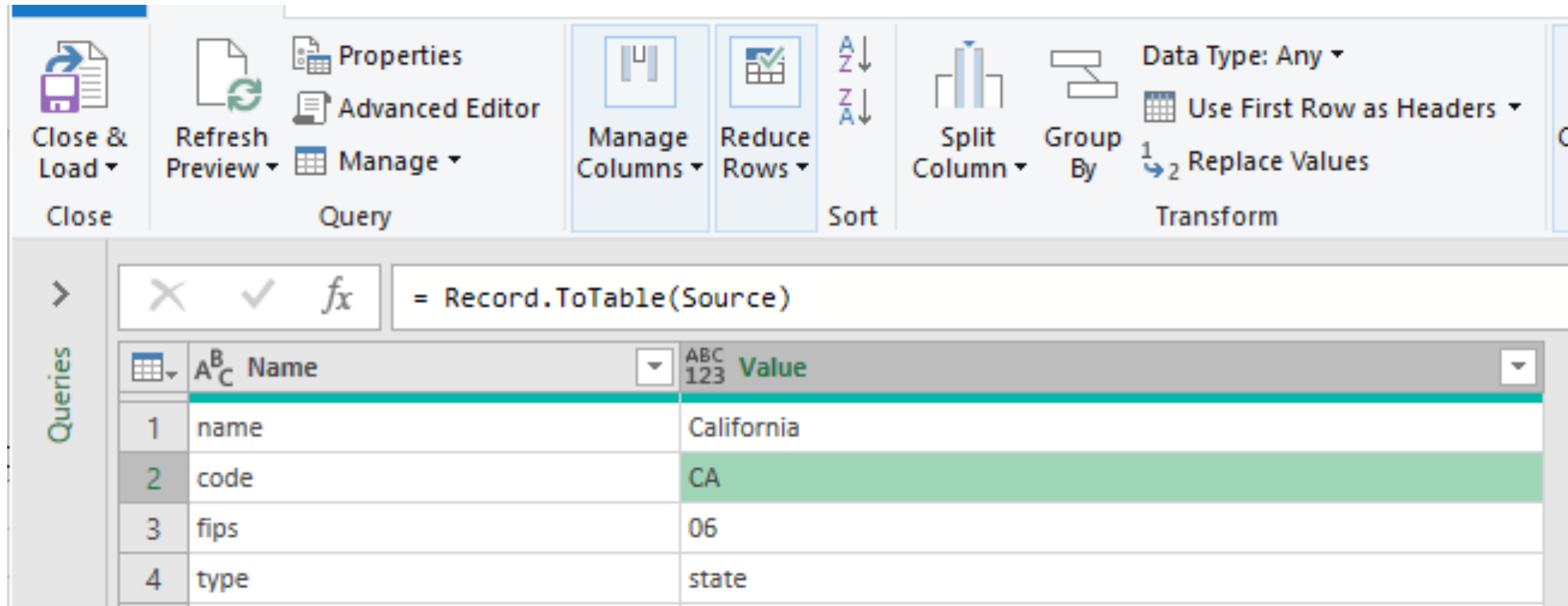
name	California
code	CA
fips	06
type	state
population	39536653
pop_year	2017
pop_source	U.S. Census Bureau, 2017 Popul...
median_household_income	67739
mhi_year	2016
mhi_source	U.S. Census Bureau, 2016 Amer...
total_prime_amount	3.9384E+11
total_prime_awards	883682
total_face_value_loan_amount	1.19972E+11

The 'Query Settings' panel shows the following settings:

- PROPERTIES**
 - Name:
 - All Properties
- APPLIED STEPS**
 - Source

Compare the request results table to the CA profile page on USAspending. Click 'Into Table'

Demo continued



The screenshot shows the Power Query ribbon with the following options:

- Close & Load (Close)
- Refresh Preview
- Query: Properties, Advanced Editor, Manage
- Manage Columns
- Reduce Rows
- Sort: A-Z, Z-A
- Split Column
- Group By
- Transform: Data Type: Any, Use First Row as Headers, Replace Values

The formula bar contains the expression: `= Record.ToTable(Source)`

The 'Queries' pane on the left shows a table with the following data:

	Name	Value
1	name	California
2	code	CA
3	fips	06
4	type	state

Select 'Close and Load'

Demo continued

	A	B
1	Name	Value
2	name	California
3	code	CA
4	fips	06
5	type	state
6	population	39536653
7	pop_year	2017
8	pop_source	U.S. Census Bureau, 2017 Population Estimate
9	median_household_income	67739
10	mhi_year	2016
11	mhi_source	U.S. Census Bureau, 2016 American Community Survey 1-Year Estimates
12	total_prime_amount	3.9384E+11
13	total_prime_awards	883682
14	total_face_value_loan_amount	1.19972E+11
15	total_face_value_loan_prime_awards	438197
16	award_amount_per_capita	9961.38

Your data is now populated in an Excel spreadsheet!

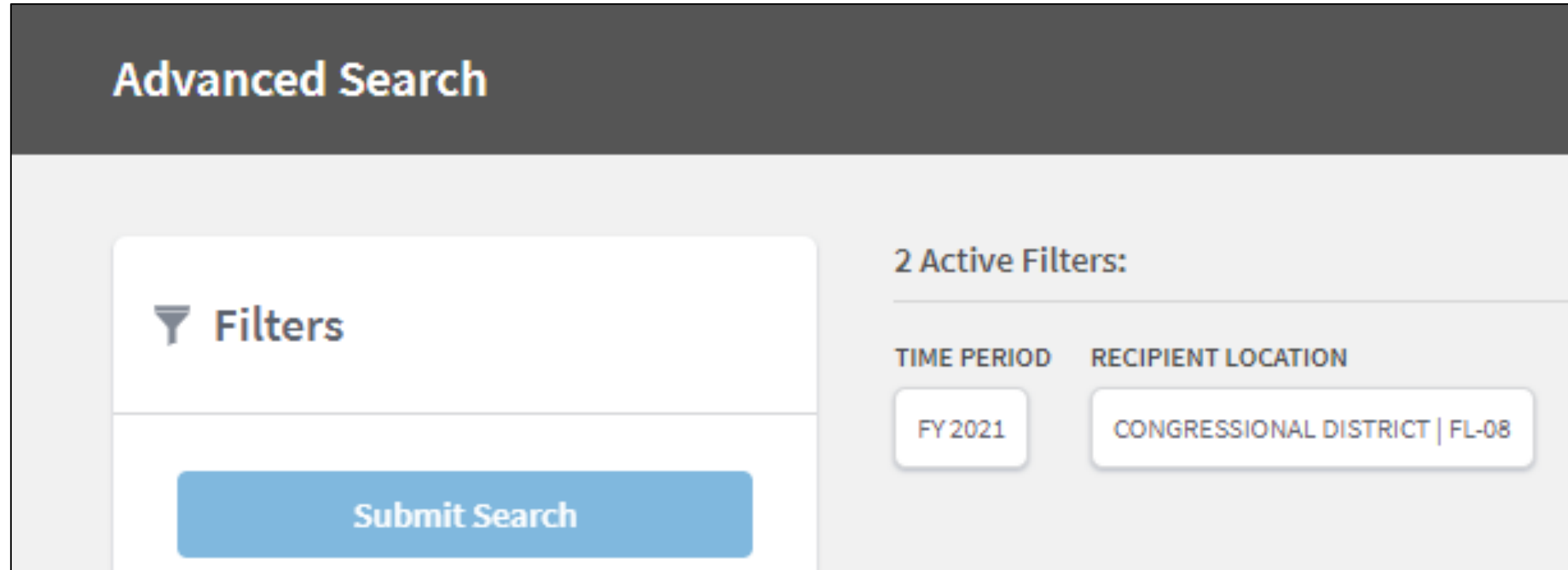
Advanced Search POST Request

How much did my Congressional District receive in a FY?

Steps:

1. Find this number using USAspending Advanced Search.
2. Inspect the page to identify which endpoints are used to display this number on the webpage.
3. Review the documentation for this endpoint.
4. Use PowerQuery to reproduce this API request in Excel.

Demo continued



The screenshot displays an "Advanced Search" interface. On the left, a white box contains a funnel icon and the text "Filters". Below this box is a blue button labeled "Submit Search". On the right, the text "2 Active Filters:" is displayed above two filter categories: "TIME PERIOD" and "RECIPIENT LOCATION". Under "TIME PERIOD", there is a button labeled "FY 2021". Under "RECIPIENT LOCATION", there is a button labeled "CONGRESSIONAL DISTRICT | FL-08".

Create an advanced search with a time period and recipient location filter.

Demo continued

The screenshot shows a web application interface with a right-click context menu open. The application has two active filters: 'TIME PERIOD' set to 'FY 2021' and 'RECIPIENT LOCATION' set to 'CONGRESSIONAL DISTRICT | FL-08'. Below the filters, there are two view options: 'TABLE' and 'TIME', with 'TIME' selected. The main heading is 'Spending by Prime A'. Below the heading, there are two tabs: 'Contracts' with a value of '5,026' and 'Contract IDVs'. At the bottom, there is a table with two columns: 'Award ID' and 'Recipient N'. The first row of the table shows 'N0001913C9999' and 'NORTHROP GE'.

2 Active Filters:

TIME PERIOD: FY 2021

RECIPIENT LOCATION: CONGRESSIONAL DISTRICT | FL-08

TABLE TIME

Spending by Prime A

Contracts 5,026 Contract IDVs

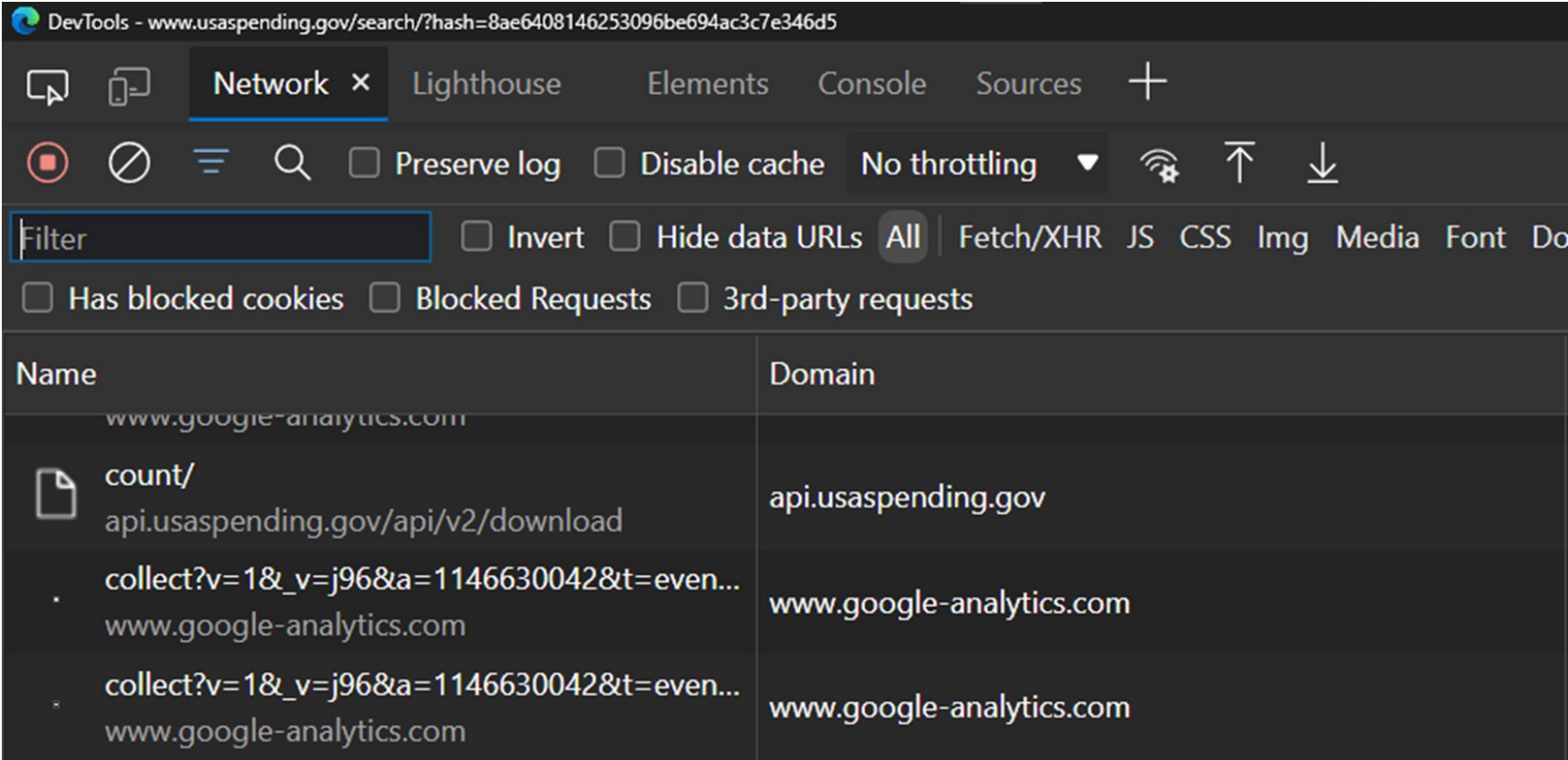
Award ID	Recipient N
N0001913C9999	NORTHROP GE

Context Menu:

- Save as (Ctrl+S)
- Print (Ctrl+P)
- Cast media to device
- Send to your devices >
- Create QR Code for this page
- Read aloud (Ctrl+Shift+U)
- Translate to English
- Add page to Collections >
- Share
- Web select (Ctrl+Shift+X)
- Web capture (Ctrl+Shift+S)
- Adobe Acrobat: PDF edit, convert, sign tools >
- Get image descriptions from Microsoft >
- View page source (Ctrl+U)
- Inspect

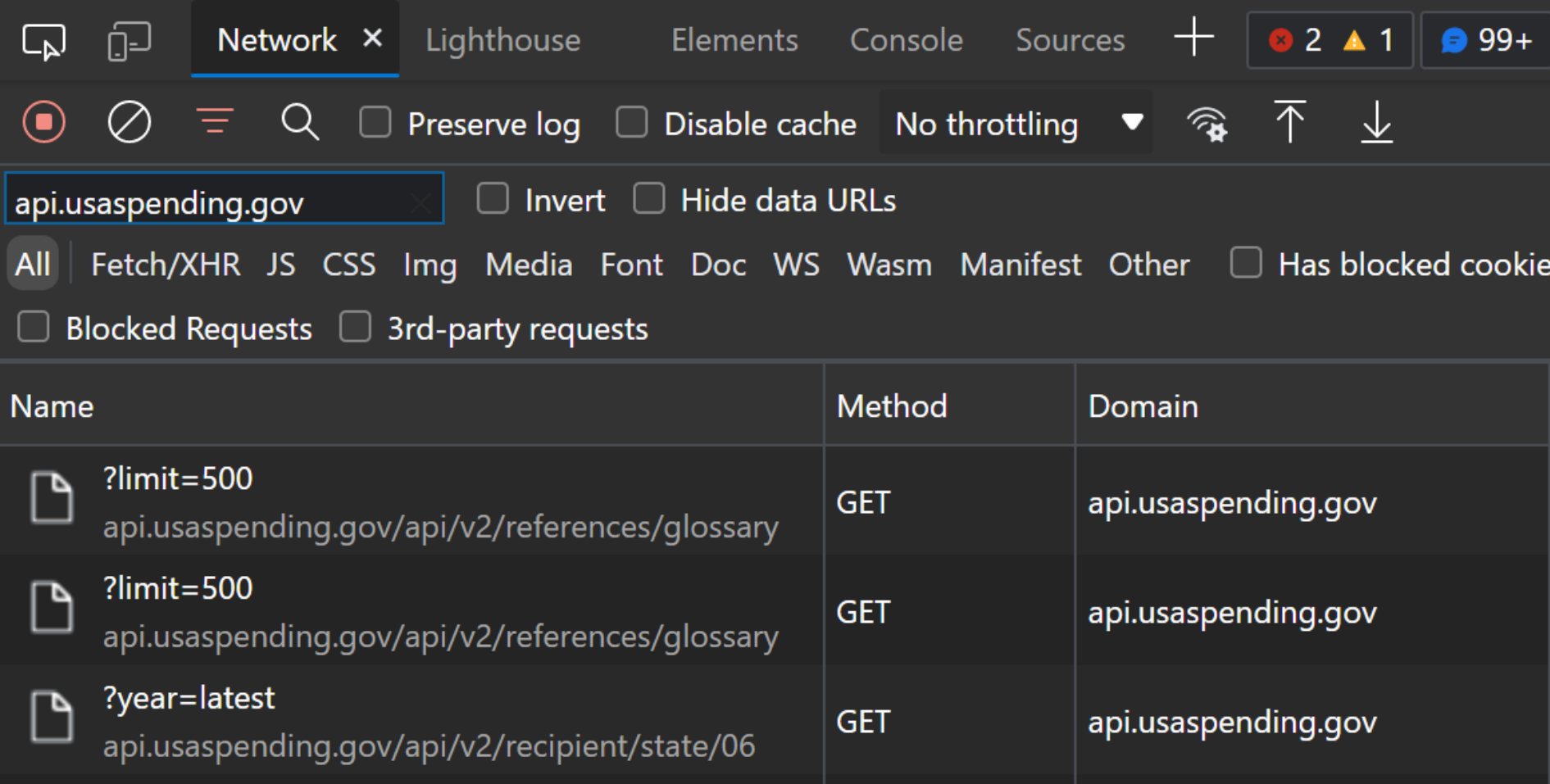
Right click the page and select Inspect.

Demo continued






Observe the network resources the browser is using to power the page.

Demo continued



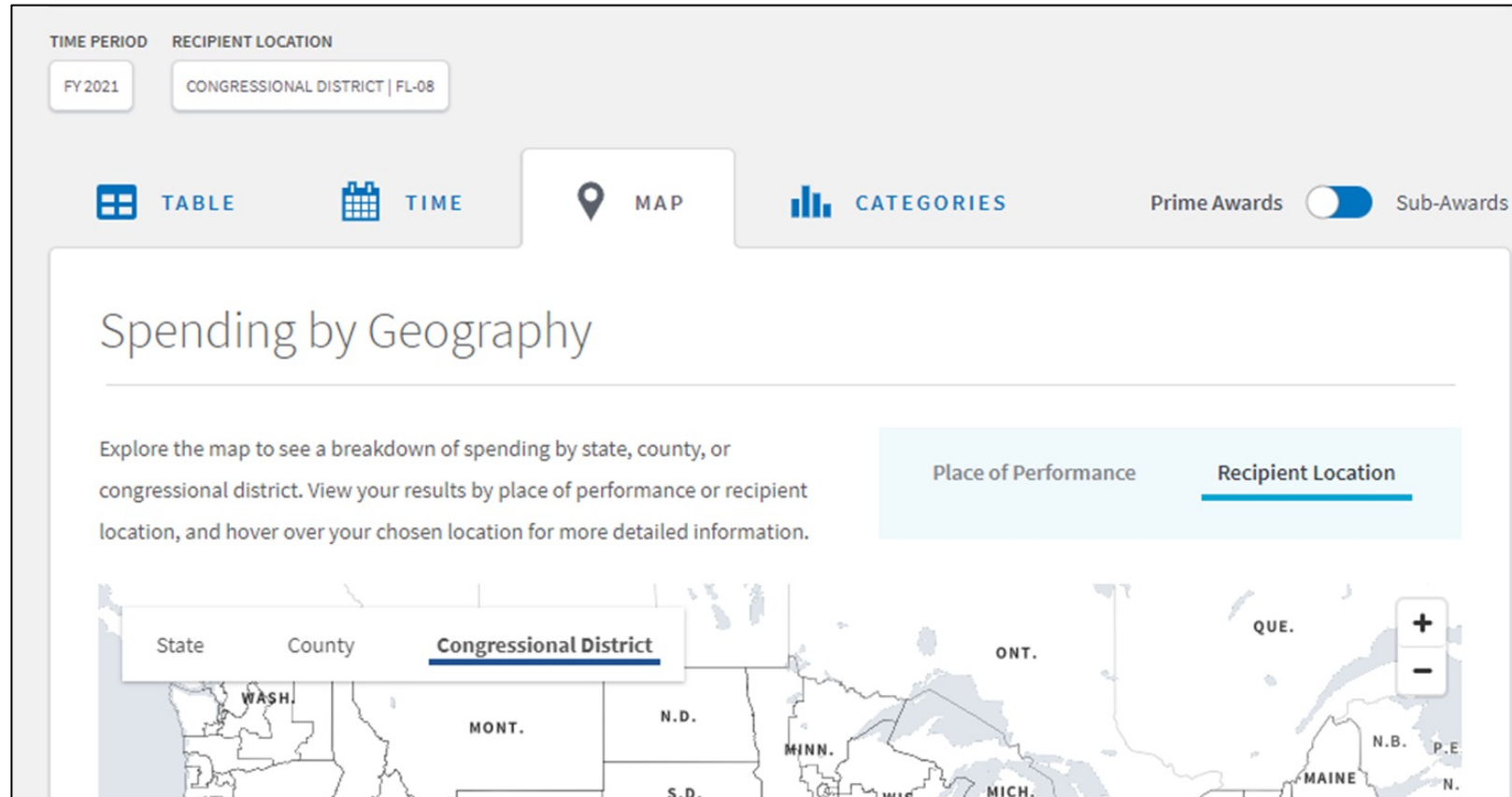
The screenshot shows the Chrome DevTools Network tab with the following configuration:

- Search filter: `api.usaspending.gov`
- Filters: Invert, Hide data URLs
- Request types: **All**, Fetch/XHR, JS, CSS, Img, Media, Font, Doc, WS, Wasm, Manifest, Other
- Advanced filters: Has blocked cookie, Blocked Requests, 3rd-party requests

Name	Method	Domain
 ?limit=500 api.usaspending.gov/api/v2/references/glossary	GET	api.usaspending.gov
 ?limit=500 api.usaspending.gov/api/v2/references/glossary	GET	api.usaspending.gov
 ?year=latest api.usaspending.gov/api/v2/recipient/state/06	GET	api.usaspending.gov

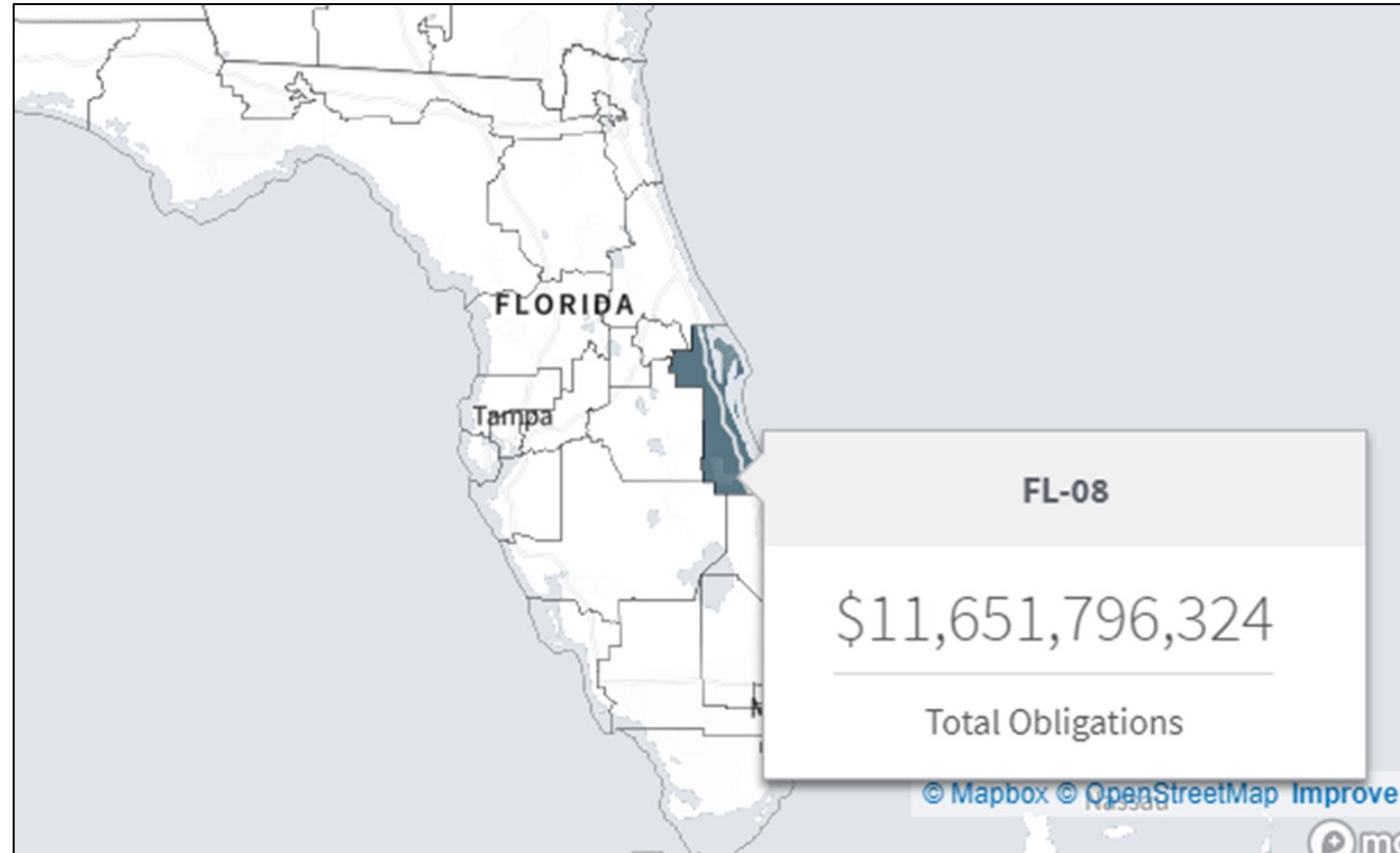
Use the filter box to only show API calls to `api.usaspending.gov`

Demo continued



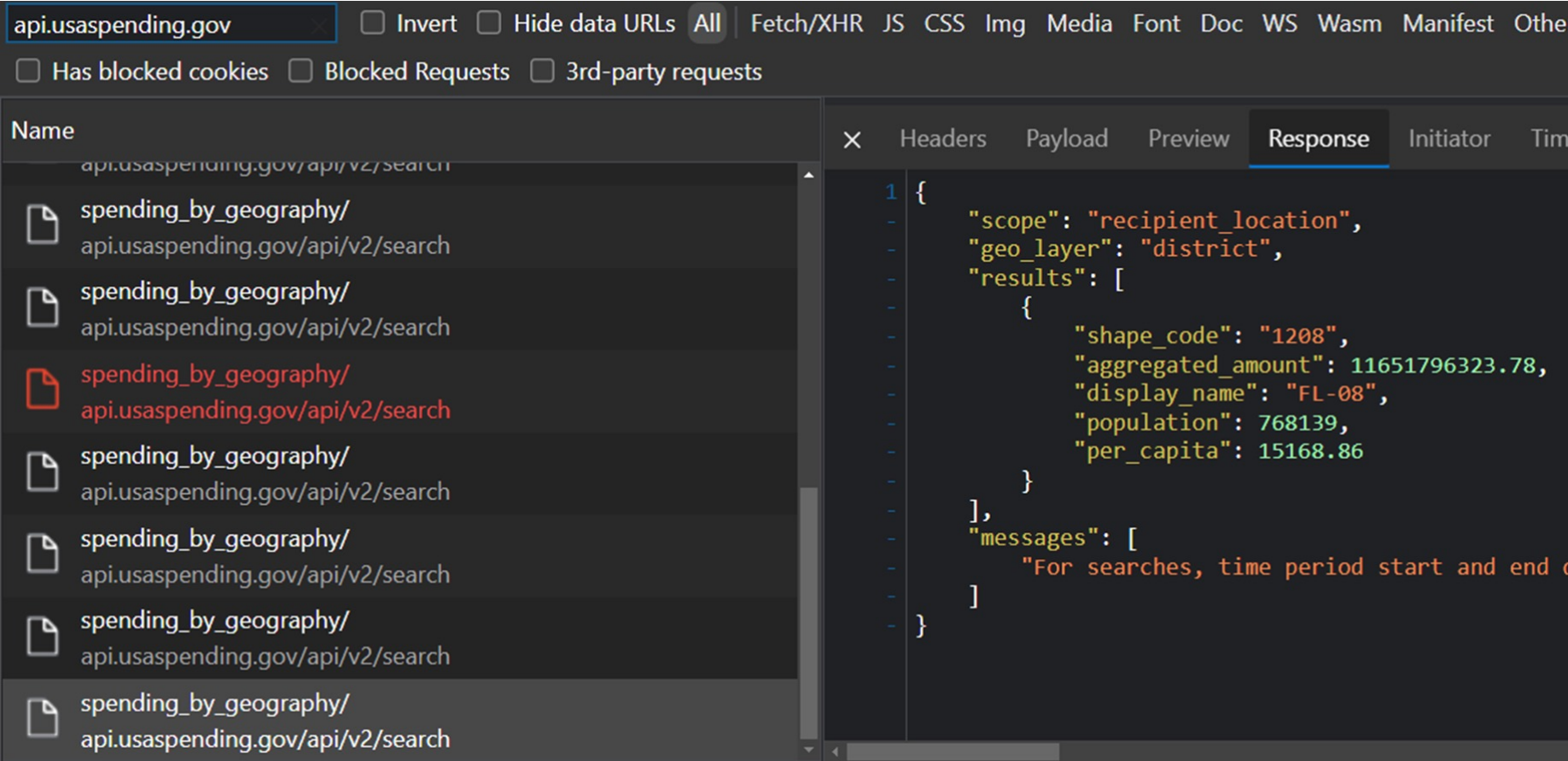
Select the map tab in USAspending Advanced Search and select Recipient Location and Congressional District

Demo continued



Zoom in and hover over the appropriate congressional district.
Notice the total obligations value.

Demo continued

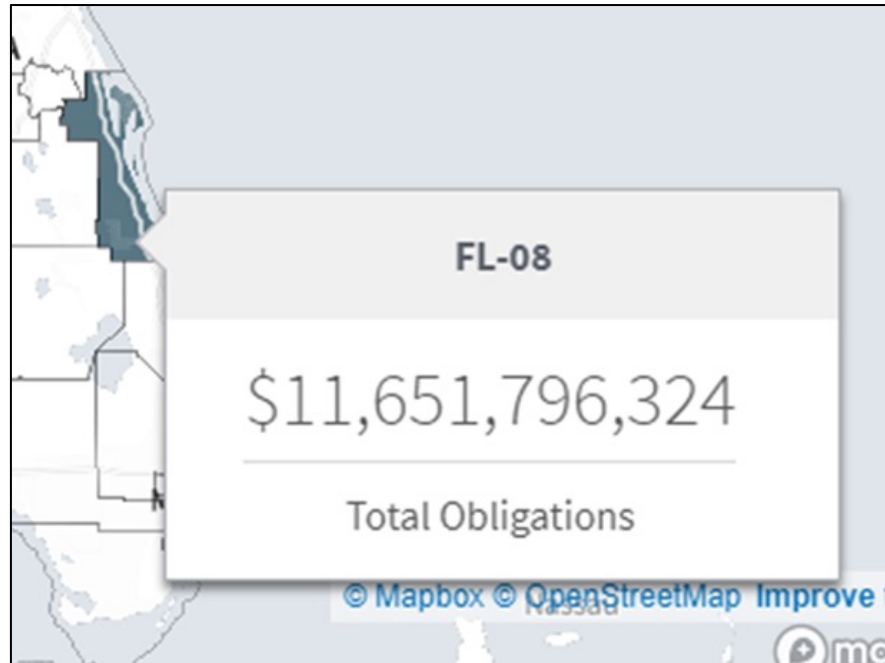


Review the Response tab for the most recent spending_by_geography API call.

Observe that the aggregated_amount value matches the map.

*Note that the API responses on USAspending are returned in JSON

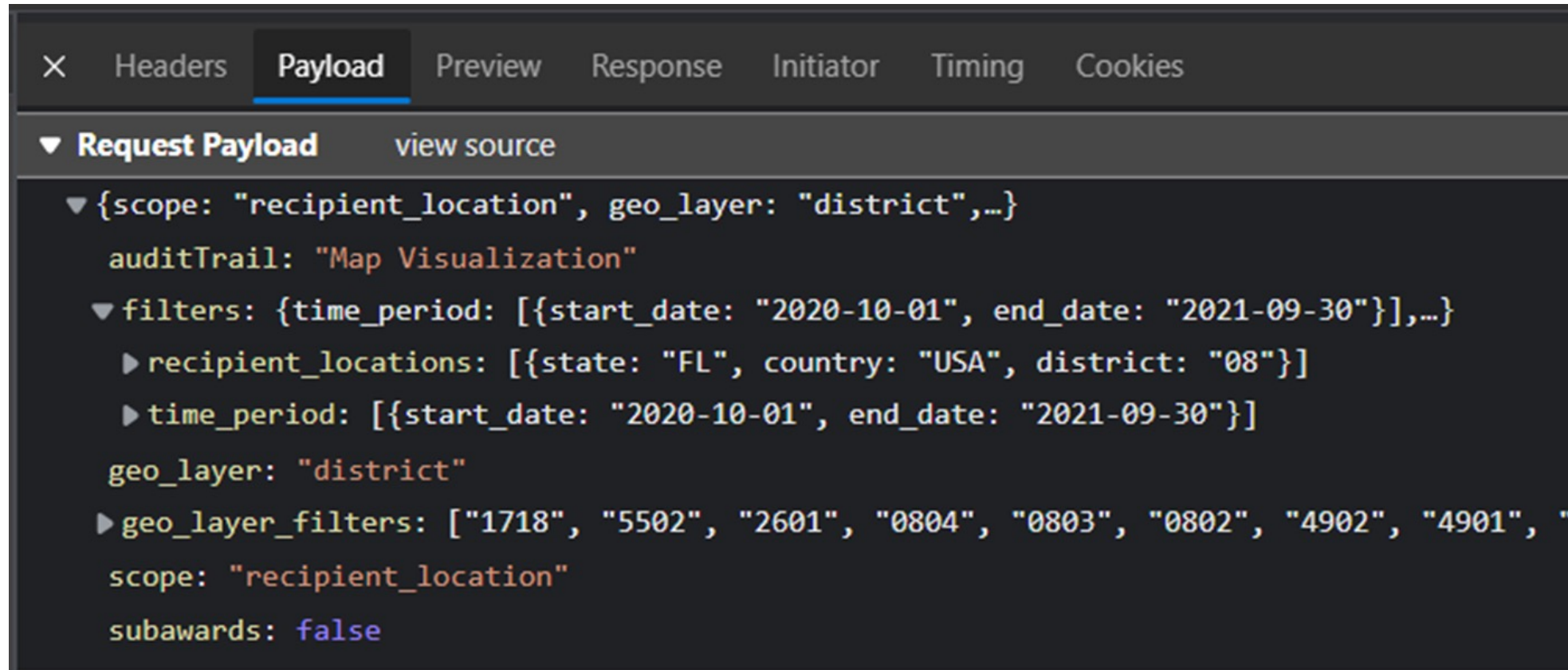
Demo continued



```
"results": [  
  {  
    "shape_code": "1208",  
    "aggregated_amount": 11651796323.78,  
    "display_name": "FL-08",  
    "population": 768139,  
    "per_capita": 15168.86  
  }  
],
```

Review the Response tab for the most recent `spending_by_geography` API call. Observe that the `aggregated_amount` value matches the map.

Demo continued

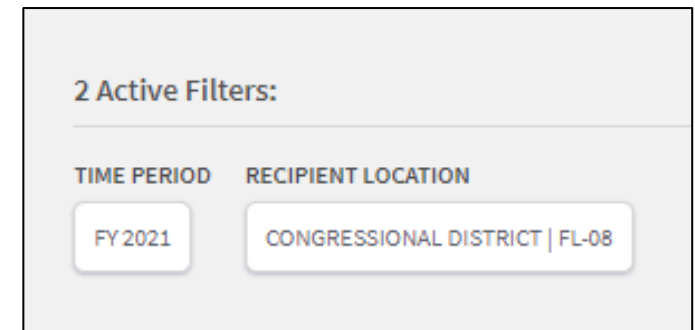


```
Request Payload view source
{scope: "recipient_location", geo_layer: "district", ...}
  auditTrail: "Map Visualization"
  filters: {time_period: [{start_date: "2020-10-01", end_date: "2021-09-30"}], ...}
    ▶ recipient_locations: [{state: "FL", country: "USA", district: "08"}]
    ▶ time_period: [{start_date: "2020-10-01", end_date: "2021-09-30"}]
  geo_layer: "district"
  ▶ geo_layer_filters: ["1718", "5502", "2601", "0804", "0803", "0802", "4902", "4901", ...]
  scope: "recipient_location"
  subawards: false
```

Review the Payload tab for the most recent `spending_by_geography` API call.
Observe that the `filters` value matches the search filters.

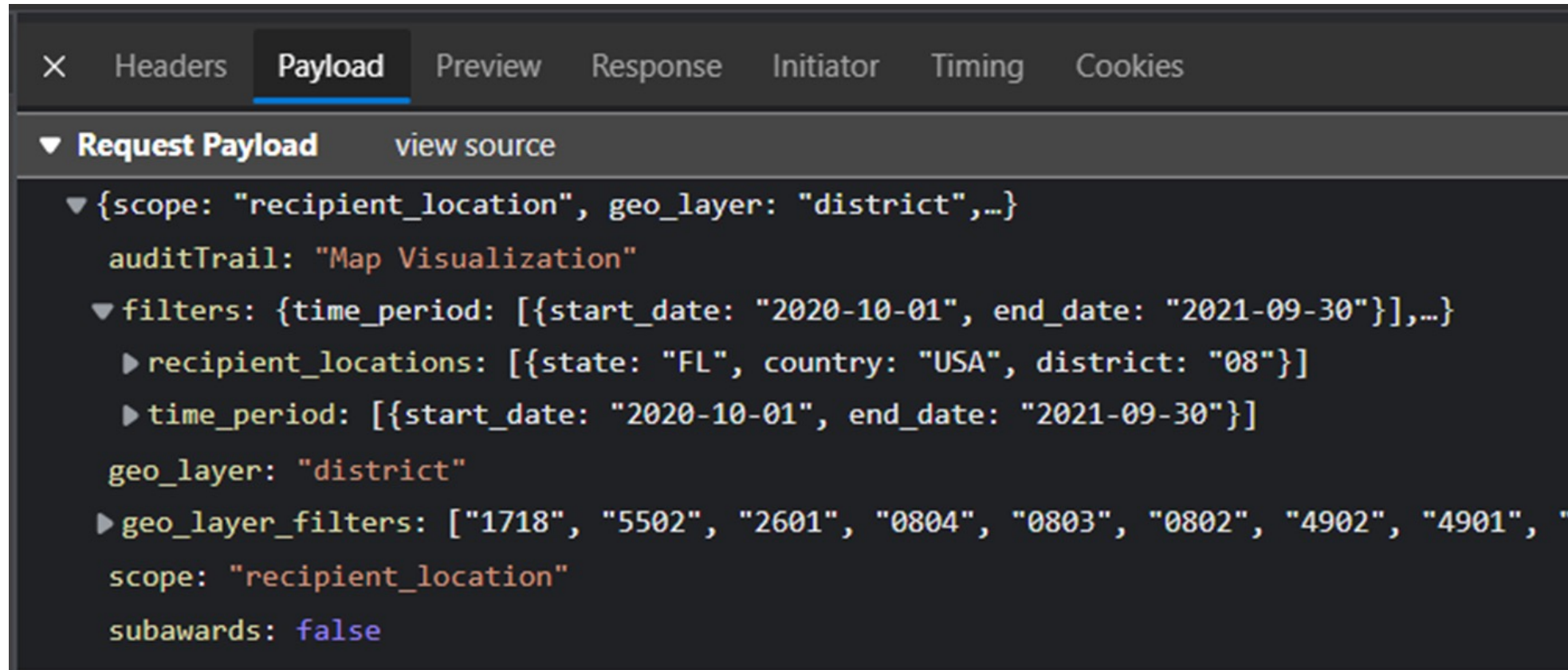
Demo continued

```
▼ filters: {time_period: [{start_date: "2020-10-01", end_date: "2021-09-30"}],...}
▼ recipient_locations: [{state: "FL", country: "USA", district: "08"}]
  ▼ 0: {state: "FL", country: "USA", district: "08"}
    country: "USA"
    district: "08"
    state: "FL"
▼ time_period: [{start_date: "2020-10-01", end_date: "2021-09-30"}]
  ▼ 0: {start_date: "2020-10-01", end_date: "2021-09-30"}
    end_date: "2021-09-30"
    start_date: "2020-10-01"
```



Review the Payload tab for the most recent `spending_by_geography` API call.
Observe that the filters value matches the search filters.

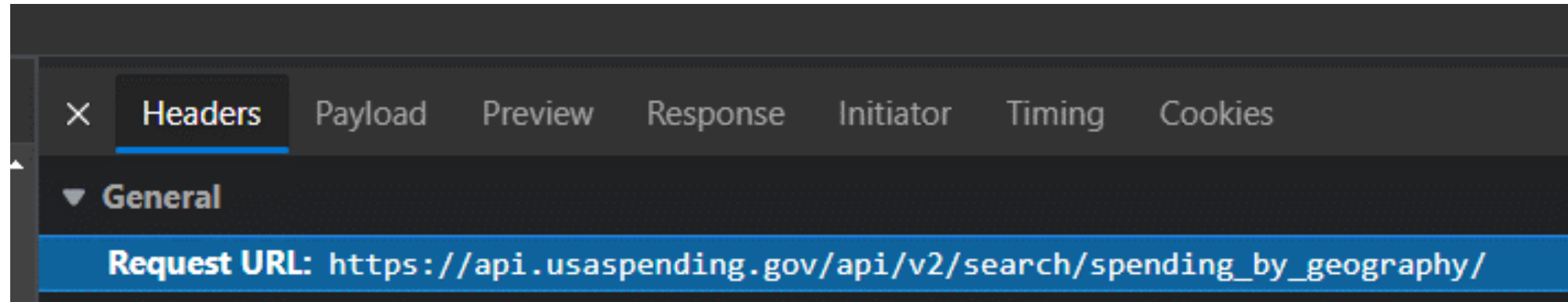
Demo continued



```
Request Payload view source
{scope: "recipient_location", geo_layer: "district", ...}
  auditTrail: "Map Visualization"
  filters: {time_period: [{start_date: "2020-10-01", end_date: "2021-09-30"}], ...}
    recipient_locations: [{state: "FL", country: "USA", district: "08"}]
    time_period: [{start_date: "2020-10-01", end_date: "2021-09-30"}]
  geo_layer: "district"
  geo_layer_filters: ["1718", "5502", "2601", "0804", "0803", "0802", "4902", "4901", ...]
  scope: "recipient_location"
  subawards: false
```

Review the Payload tab for the most recent spending_by_geography API call. Observe that the filters value matches the search filters.

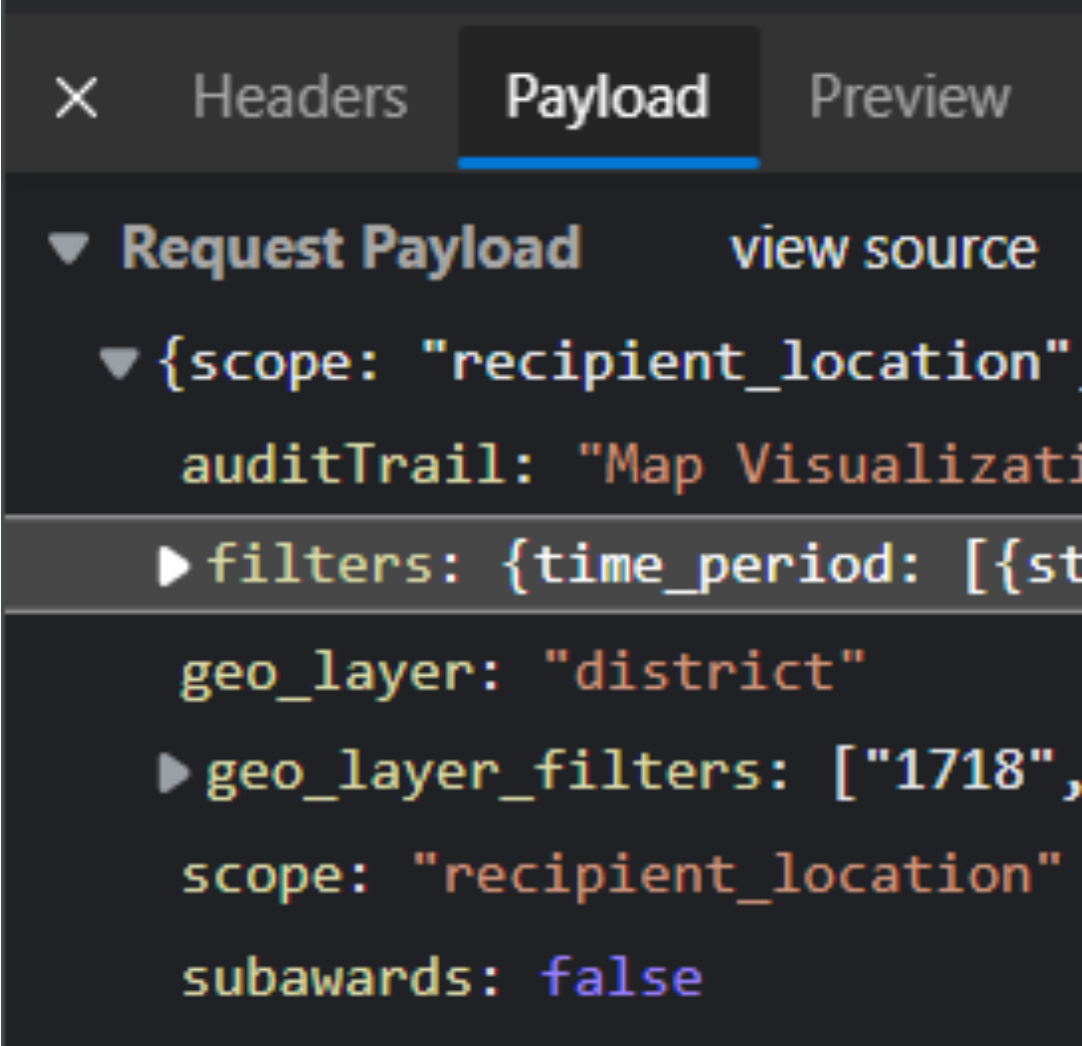
Demo continued



Select the Headers tab and observe the Request URL value.
Navigate to the URL and click “Documentation for this endpoint can be found here.”

Demo continued

- Attributes (object)
 - `filters` (required, AdvancedFilterObject)
 - `subawards` (optional, boolean) True when you want to group by Subawards instead of Awards. Defaulted to False.
 - `scope` (required, enum[string]) When fetching transactions, use the primary place of performance or recipient location
 - Members
 - `place_of_performance`
 - `recipient_location`
 - `geo_layer` (required, enum[string]) Set the type of areas in the response
 - Members
 - `state`
 - `county`
 - `district`
 - `geo_layer_filters` (optional, array[string])



The screenshot shows a web application interface with three tabs: 'Headers', 'Payload', and 'Preview'. The 'Payload' tab is selected and highlighted with a blue underline. Below the tabs, there is a section titled 'Request Payload' with a 'view source' link. The payload is a JSON object displayed in a dark-themed code editor. The visible portion of the JSON is:

```
{scope: "recipient_location", auditTrail: "Map Visualizati", filters: {time_period: [{st}, geo_layer: "district", geo_layer_filters: ["1718", scope: "recipient_location", subawards: false
```

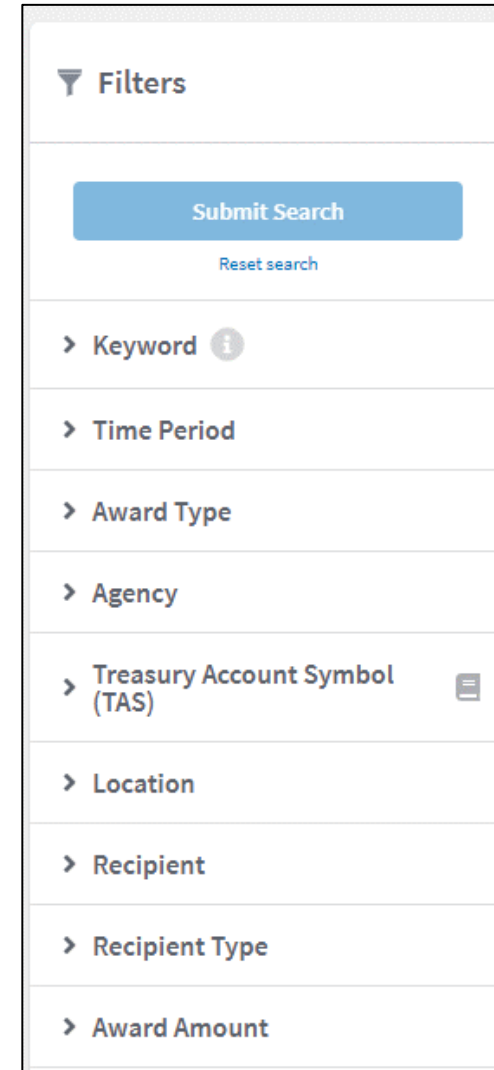
Compare the documentation attributes section and the Payload tab. Observe the keys in the payload match the attribute bullets (filters, subawards, scope, etc.). Notice the filters attribute takes an AdvancedFiterObject

Demo continued

```
AdvancedFilterObject (object)

- keywords : transport (optional, array[string])
- time_period (optional, array[TimePeriodObject], fixed-type)
- place_of_performance_scope (optional, enum[string])
  - Members
    - domestic
    - foreign
- place_of_performance_locations (optional, array[LocationObject], fixed-type)
- agencies (optional, array[AgencyObject], fixed-type)
- recipient_search_text : Hampton (optional, array[string])
  - Text searched across a recipient's name, UEI, and DUNS
- recipient_id (optional, string) A hash of recipient DUNS, name, and level. A unique identifier for recipients, used for profile page urls.
- recipient_scope (optional, enum[string])
  - Members
    - domestic
    - foreign
- recipient_locations (optional, array[LocationObject], fixed-type)

```



Compare the AdvancedFilterObject documentation and the Advanced Search filter options. AdvancedFilterObject powers advanced search, it is very useful!

Demo continued

`time_period` (optional, array[TimePeriodObject], fixed-type)

TimePeriodObject (object)

- `start_date` : `2017-10-01` (required, string) Currently limited to an earliest date of `2007-10-01` (FY2008). For data going back to `2000-10-01` (FY2001), use either the Custom Award Download feature on the website or one of our `download` or `bulk_download` API endpoints.
- `end_date` : `2018-09-30` (required, string) Currently limited to an earliest date of `2007-10-01` (FY2008). For data going back to `2000-10-01` (FY2001), use either the Custom Award Download feature on the website or one of our `download` or `bulk_download` API endpoints.
- `date_type` (optional, enum[string])
 - Members
 - `action_date`
 - `last_modified_date`

`recipient_locations` (optional, array[LocationObject], fixed-type)

LocationObject (object)

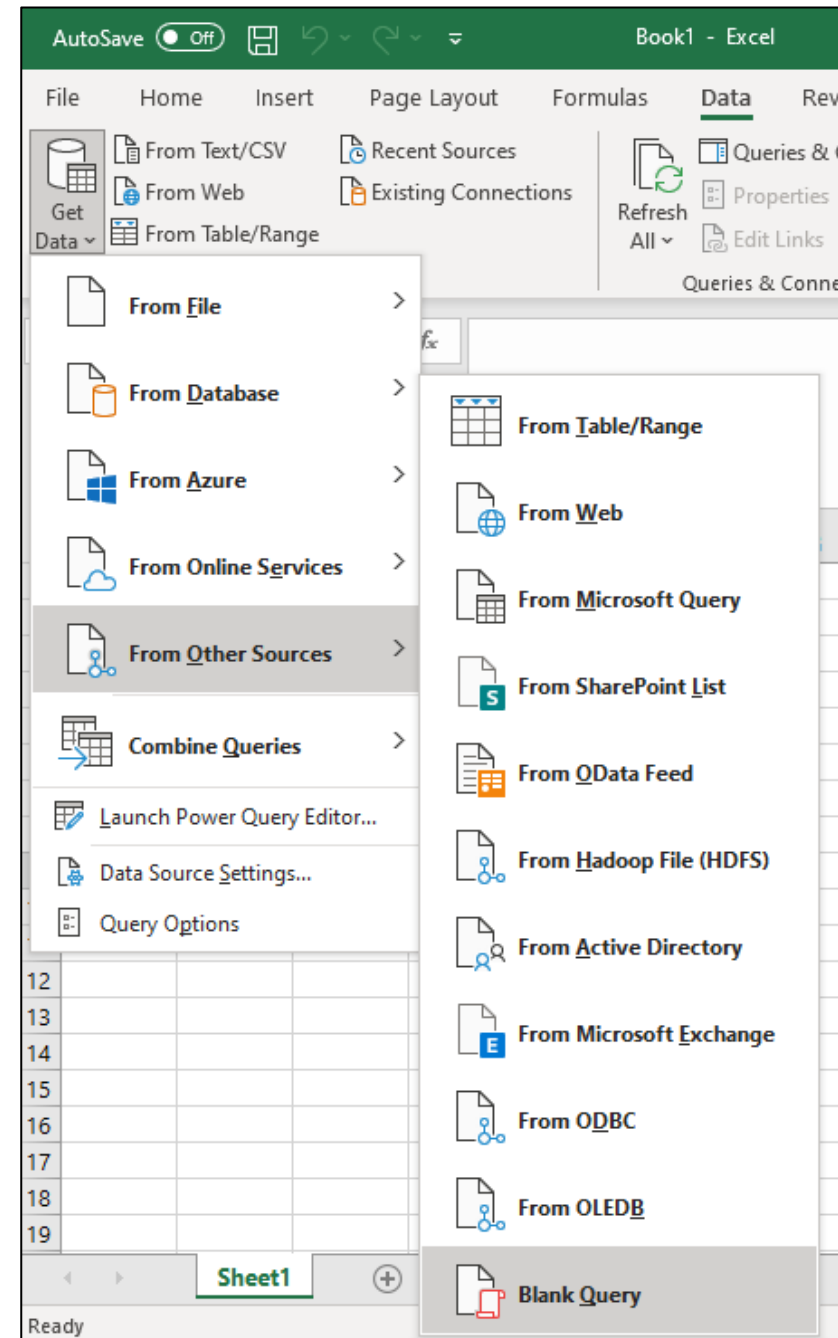
- `country` : `USA` (required, string)
- `state` : `VA` (optional, string)
- `county` (optional, string)
- `city` (optional, string)
- `district` (optional, string)
- `zip` (optional, string)

Observe that the `time_period` filter takes a `TimePeriodObject` and the `recipient_locations` filter takes a `LocationObject`.

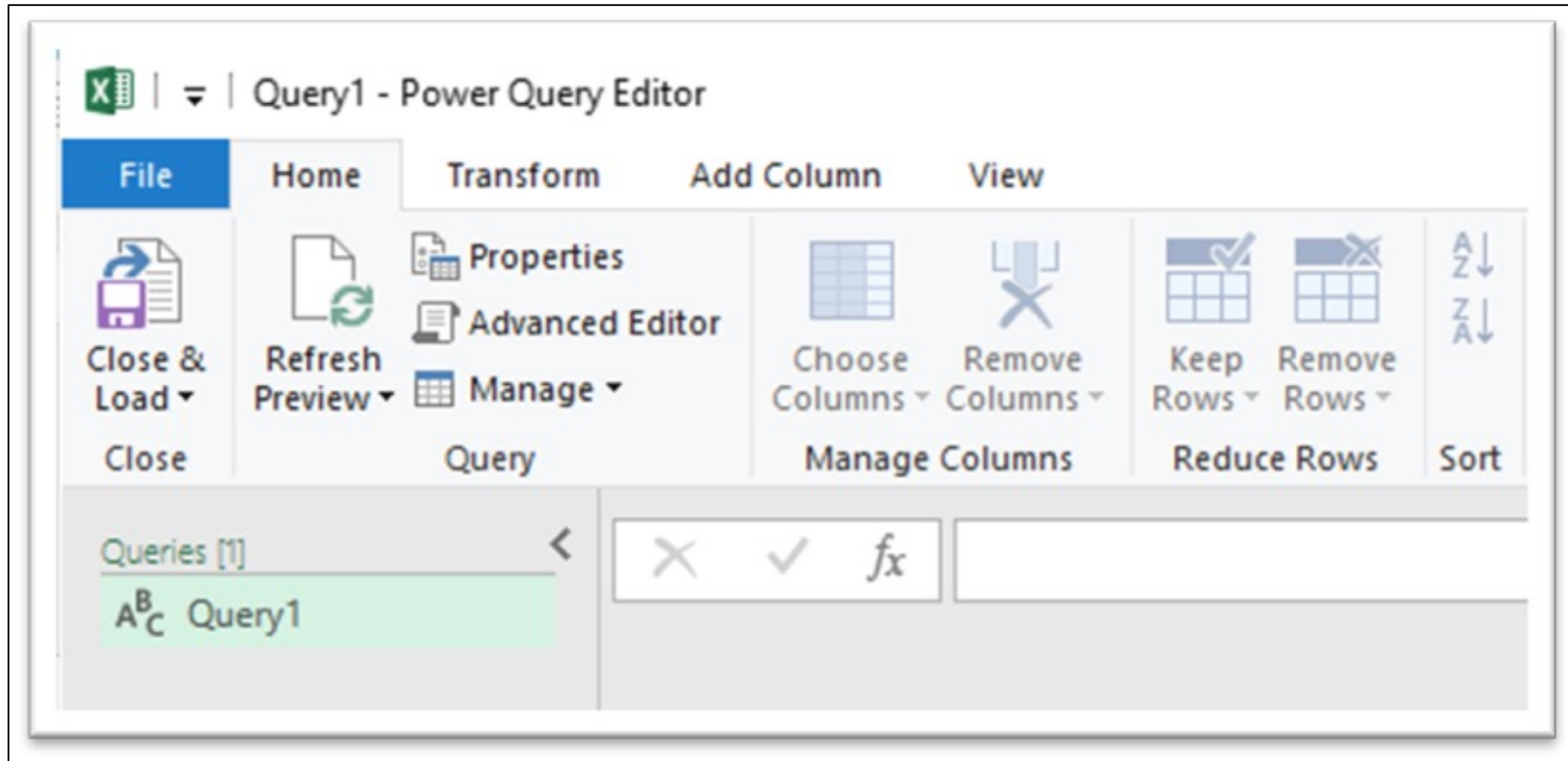
Demo continued

You can use PowerQuery in Excel to import data using POST API requests.

In a new Excel workbook select
Data > Get Data > From Other Sources > Blank Query



Demo continued



In the PowerQuery Editor window, select 'Advanced Editor'

Demo continued

Query1

```
1 let
2   url = "https://api.usaspending.gov/api/v2/search/spending_by_geography/",
3   body = "{
4     ""scope"": ""recipient_location"",
5     ""geo_layer"": ""district"",
6     ""filters"": {
7       ""time_period"": [
8         {
9           ""start_date"": ""2020-10-01"",
10          ""end_date"": ""2021-09-30""
11        }
12      ],
13     ""recipient_locations"": [
14       {
15         ""state"": ""FL"",
16         ""country"": ""USA"",
17         ""district"": ""08""
18       }
19     ]
20   }",
21   Source = Json.Document(Web.Contents(url,[Content=Text.ToBinary(body),Headers={"Content-Type":"application/json"}]))
23 in
24   Source
```

Replace the default code with the code displayed above also available on the link below)

Notes: <https://github.com/fedspendingtransparency/usaspending-api/wiki/API-Usage---Power-Query-Example>

Demo continued

Query1

```
1 let
2   url = "https://api.usaspending.gov/api/v2/search/spending_by_geography/",
3   body = "{
4     ""scope"": ""recipient_location"",
5     ""geo_layer"": ""district"",
6     ""filters"": {
7       ""time_period"": [
8         {
9           ""start_date"": ""2020-10-01"",
10          ""end_date"": ""2021-09-30""
11        }
12      ],
13      ""recipient_locations"": [
14        {
15          ""state"": ""FL"",
16          ""country"": ""USA"",
17          ""district"": ""08""
18        }
19      ]
20    }
21  }",
22   Source = Json.Document(Web.Contents(url,[Content=Text.ToBinary(body),Headers=[#"Content-Type"="application/json"]]))
23 in
24   Source
```

PowerQuery handles POST requests in a particular way. For example, lines 1, and 22-24 as well as the double double-quotes.

Notes: <https://github.com/fedspendingtransparency/usaspending-api/wiki/API-Usage---Power-Query-Example>

Demo continued

```
url = "https://api.usaspending.gov/api/v2/search/spending_by_geography/",
body = "{
  ""scope"": ""recipient_location"",
  ""geo_layer"": ""district"",
  ""filters"": {
    ""time_period"": [
      {
        ""start_date"": ""2020-10-01"",
        ""end_date"": ""2021-09-30""
      }
    ],
    ""recipient_locations"": [
      {
        ""state"": ""FL"",
        ""country"": ""USA"",
        ""district"": ""08""
      }
    ]
  }
}"
```

```
Source = Json.Document(Web.Contents(url).[Content=
```

The image shows two screenshots of a web browser's developer tools. The top screenshot shows the 'Headers' tab, where the 'Request URL' is highlighted as 'https://api.usaspending.gov/api/v2/search/spending_by_geography/'. The bottom screenshot shows the 'Payload' tab, where the 'Request Payload' is displayed as a JSON object: {scope: "recipient_location", geo_layer: "district", ...}. The payload is expanded to show details like 'auditTrail: "Map Visualization"', 'filters' (with time period and recipient locations), 'geo_layer: "district"', 'geo_layer_filters' (a list of district IDs), 'scope: "recipient_location"', and 'subawards: false'.

Compare the code url variable value and the Headers tab Request URL value. Compare the code body variable value and the Payload tab Request Payload value.

Notes: <https://github.com/fedspendingtransparency/usaspending-api/wiki/API-Usage---Power-Query-Example>

Demo continued

```
22 | Source = Json.Document(Web.Contents(url,[Content=Text.ToBinary(body),Headers=[#"Content-Type"="application/json"]]))
23 | in
24 | Source
```

✓ No syntax errors have been detected.

Done

Cancel

Lines 22 – 24 are required to make POST API requests in PowerQuery.

Notice the code has no syntax errors.

Click Done.

Notes: <https://github.com/fedspendingtransparency/usaspending-api/wiki/API-Usage---Power-Query-Example>

Demo continued

The screenshot shows a query tool interface. On the left, a table displays query results:

Property	Value
scope	recipient_location
geo_layer	district
results	List
messages	List

On the right, the 'Query Settings' panel is open, showing the following details:

- NAME:** Query1
- APPLIED STEPS:** url, body, Source (selected)

The screenshot shows the 'Response' tab of a tool, displaying the following JSON data:

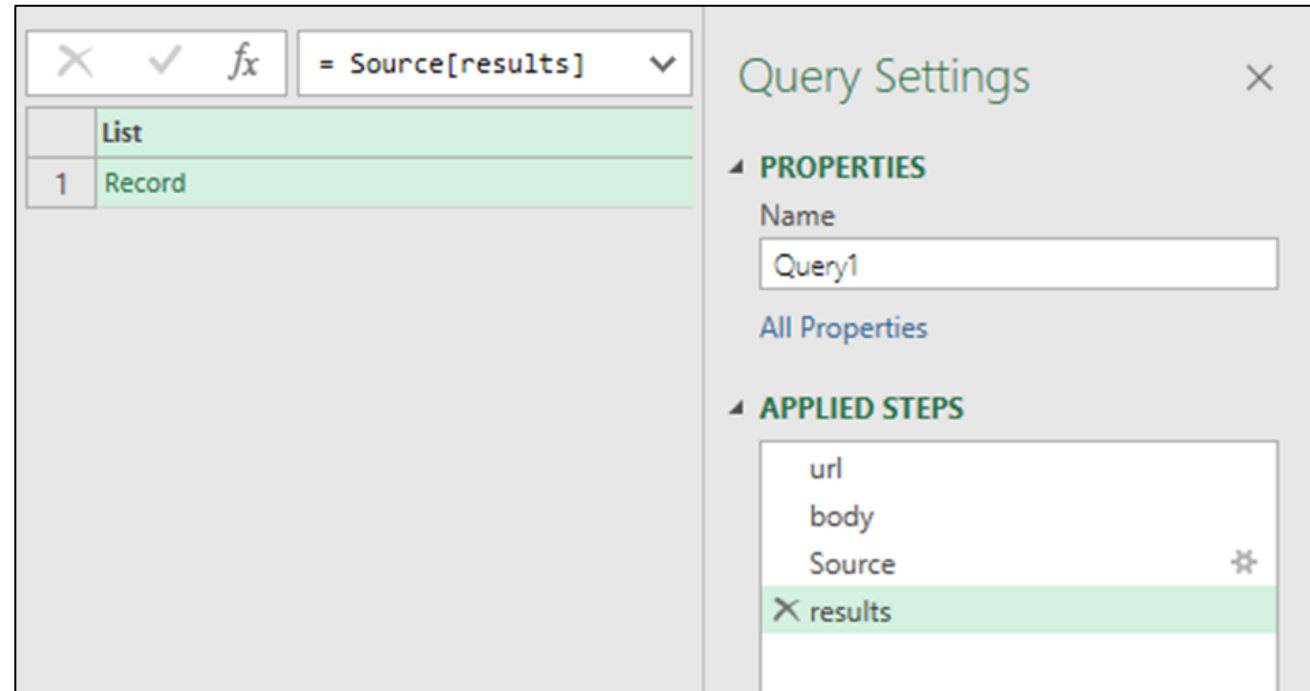
```
{
  "scope": "recipient_location",
  "geo_layer": "district",
  "results": [
    {
      "shape_code": "1208",
      "aggregated_amount": 11651796323.78,
      "display_name": "FL-08",
      "population": 768139,
      "per_capita": 15168.86
    }
  ],
  "messages": [
    "For searches, time period start and end da"
  ]
}
```

Observe the query results table and Query Settings panel.

Compare the table with the Response tab.

Notice that the results record is a List. Click on the List to drill down.

Demo continued



The screenshot shows a data tool interface. On the left, a table with a header row 'List' and one data row '1 Record' is displayed. The table has a formula bar above it containing '= Source[results]'. On the right, a 'Query Settings' panel is open, showing 'Name' as 'Query1' and 'APPLIED STEPS' as a list containing 'url', 'body', 'Source', and 'results' (highlighted).

	List
1	Record

✕ ✓ fx = Source[results] ▼

Query Settings ✕

▲ PROPERTIES

Name
Query1

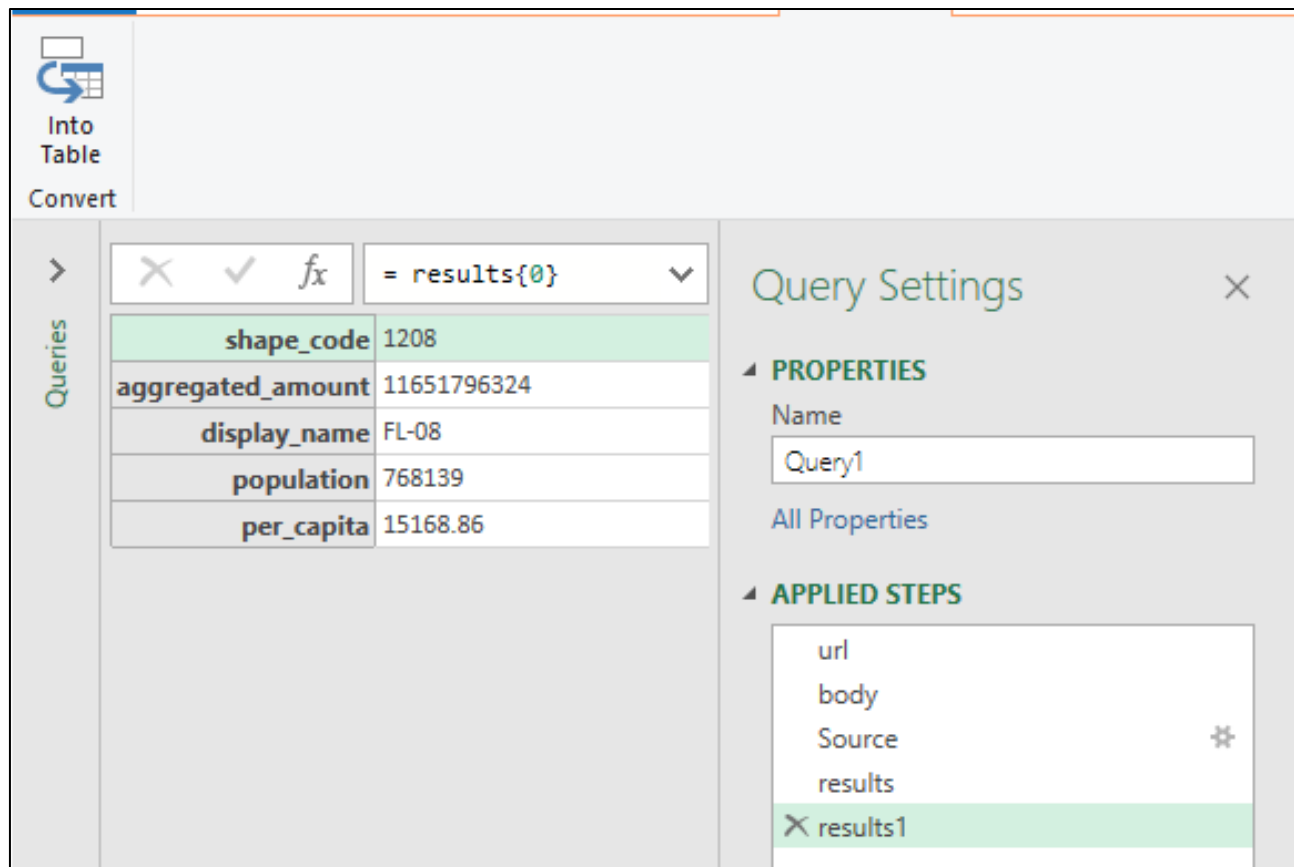
All Properties

▲ APPLIED STEPS

url
body
Source *
✕ results

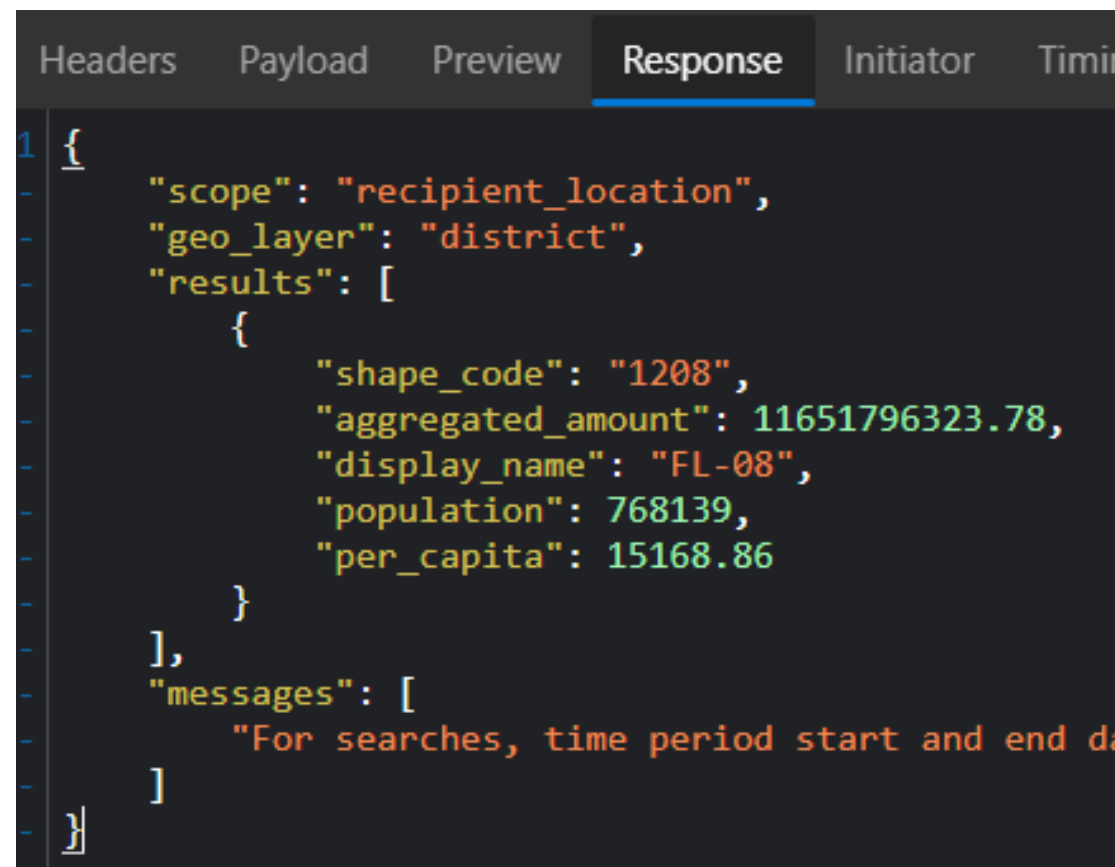
The List has one item, a Record.
Notice the updated APPLIED STEPS in the Query Settings panel.
Click the Record to drill down.

Demo continued



The screenshot shows a web application interface. On the left, there is a 'Queries' panel with a table of data. The table has five rows and two columns: a property name and its value. The first row is highlighted in green. To the right of the table is a 'Query Settings' panel. It has a 'Name' field containing 'Query1' and a list of 'APPLIED STEPS' including 'url', 'body', 'Source', 'results', and 'results1' (which is highlighted in green). At the top left of the interface, there is a button labeled 'Into Table' with a grid icon and the text 'Convert' below it.

Property	Value
shape_code	1208
aggregated_amount	11651796324
display_name	FL-08
population	768139
per_capita	15168.86



The screenshot shows a web application interface with a dark theme. The 'Response' tab is selected, displaying a JSON object. The JSON object has a 'scope' of 'recipient_location', a 'geo_layer' of 'district', and a 'results' array. The 'results' array contains one object with the following properties: 'shape_code' (1208), 'aggregated_amount' (11651796323.78), 'display_name' (FL-08), 'population' (768139), and 'per_capita' (15168.86). There is also a 'messages' array with one message: 'For searches, time period start and end da...'. The JSON is syntax-highlighted with colors.

```
{
  "scope": "recipient_location",
  "geo_layer": "district",
  "results": [
    {
      "shape_code": "1208",
      "aggregated_amount": 11651796323.78,
      "display_name": "FL-08",
      "population": 768139,
      "per_capita": 15168.86
    }
  ],
  "messages": [
    "For searches, time period start and end da..."
  ]
}
```

Finally, our data! It should match the Response tab and the map view.

Notice the updated applied steps.

Click 'Into Table'.

Demo continued

The screenshot shows the Power BI Desktop interface. The ribbon at the top includes the following groups: Close & Load, Query (with Refresh Preview and Manage), Manage Columns, Reduce Rows, Sort (with A-Z and Z-A), Split Column, Group By, Data Type: Text, Use First Row as Headers, Replace Values, and Combine. Below the ribbon, the 'Queries' pane on the left shows a table with the following data:

	Name	Value
1	shape_code	1208
2	aggregated_amount	11651796324
3	display_name	FL-08
4	population	768139
5	per_capita	15168.86

The 'Query Settings' pane on the right shows the following sections:

- PROPERTIES**
 - Name: recip_loc_cd
- APPLIED STEPS**
 - url
 - body
 - Source *
 - results
 - results1
 - Converted to Table

Once again, notice the updated applied steps.
Click 'Advanced Editor' again.

Demo continued

```
22 Source = Json.Document(Web.Contents(url,[Content=Text.ToBinary(body),Headers=[#"Content-Type"="application/json"]]))
23 in
24 Source
```

✓ No syntax errors have been detected.

Done

Cancel

```
22 Source = Json.Document(Web.Contents(url,[Content=Text.ToBinary(body),Headers=[#"Content-Type"="application/json"]])),
23 results = Source[results],
24 results1 = results{0},
25 #"Converted to Table" = Record.ToTable(results1)
26 in
27 #"Converted to Table"
```

Notice that the code has been updated to reflect the additional steps.

Use this code to automate those steps in the future.

Don't change the added code. Click 'Done'.

Demo continued

The screenshot shows the Power Query ribbon with the following tabs: File, Home, Transform, Add Column, and View. The 'Close & Load' button is highlighted in the File tab. The 'Advanced Editor' is open, showing the formula `= Record.ToTable(results1)`. Below the formula bar, the 'Queries' pane displays a table with the following data:

	Name	Value
1	shape_code	1208
2	aggregated_amount	11651796324
3	display_name	FL-08
4	population	768139
5	per_capita	15168.86

	A	B
1	Name	Value
2	shape_code	1208
3	aggregated_amount	11651796324
4	display_name	FL-08
5	population	768139
6	per_capita	15168.86

Click 'Close & Load' to load results into a spreadsheet.

Additional Exercises



COVID-19 Exercise With Steps

How much did my Congressional District receive in COVID-19 funding?

Steps:

1. Find this number using the USAspending COVID profile page.
2. Inspect the page to identify which endpoint is used to display this number on the webpage.
3. Review the documentation for this endpoint.
4. Use PowerQuery to reproduce this API request in Excel.

COVID Profile Page vs Advanced Search DEFC

Agencies report transaction-level award data and COVID-19 DEFC award spending data through different systems.

USAspending links the award data reported through these different systems to create a more comprehensive picture of government spending.

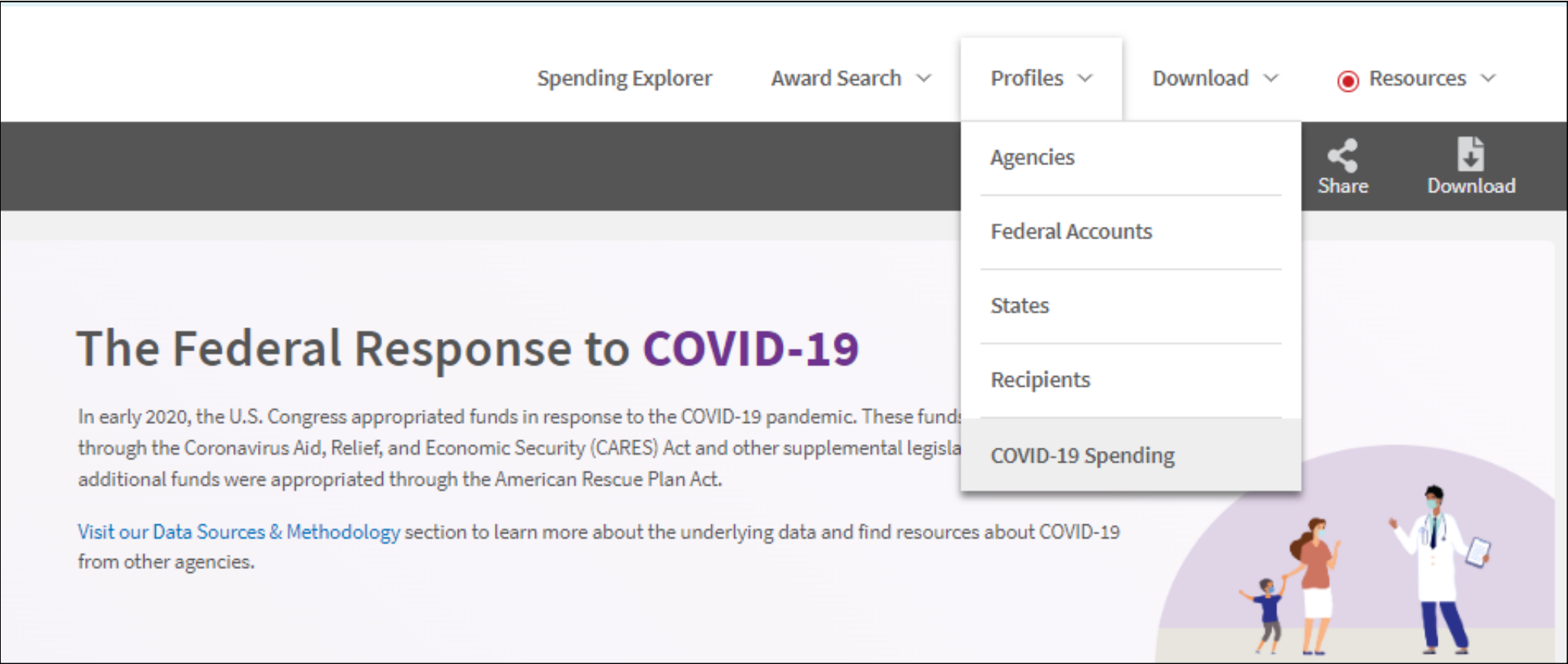
The USAspending Advanced Search tool primarily uses transaction-level award data to filter and present search results. However, since DEFC information is not available at the transaction level, summary stats (in the Time tab, Map tab, and Categories tab) for Advanced Search results with a DEFC filter are an approximation.

We built the COVID-19 Spending profile page to present up to date and detailed information on COVID spending.

The COVID-19 Spending profile page uses a different set of endpoints compared to Advanced Search.

For more information on COVID spending data sources and methodology, please see: <https://www.usaspending.gov/disaster/covid-19/data-sources>

Demo continued



Navigate to the USAspending COVID-19 Spending profile page.

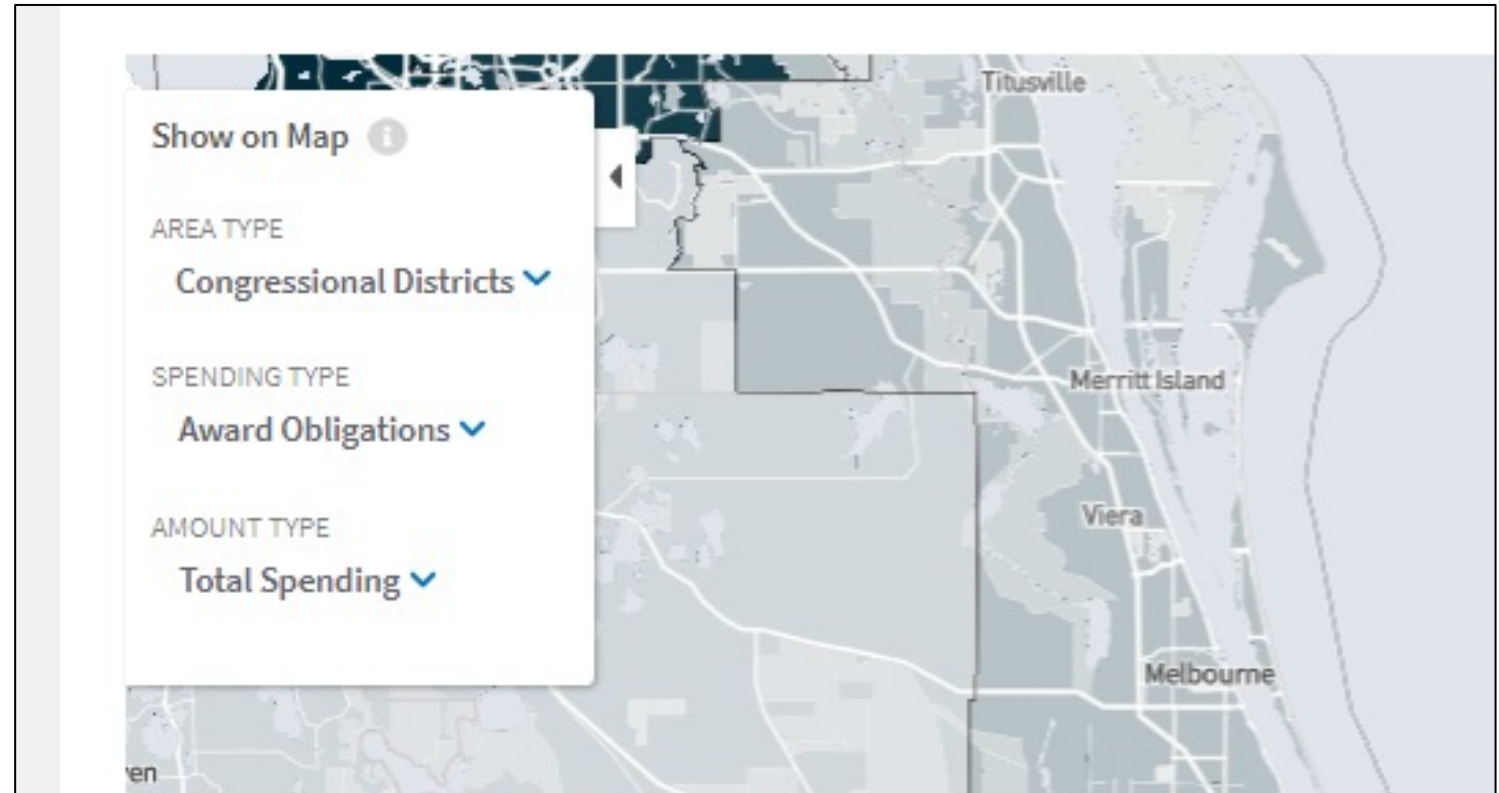
Demo continued

Scroll down to the Award Spending by Recipient Map.

Inspect the page

Configure the map:

- Area Type: Congressional District
- Spending Type: Award Obligations
- Amount Type: Total Spending

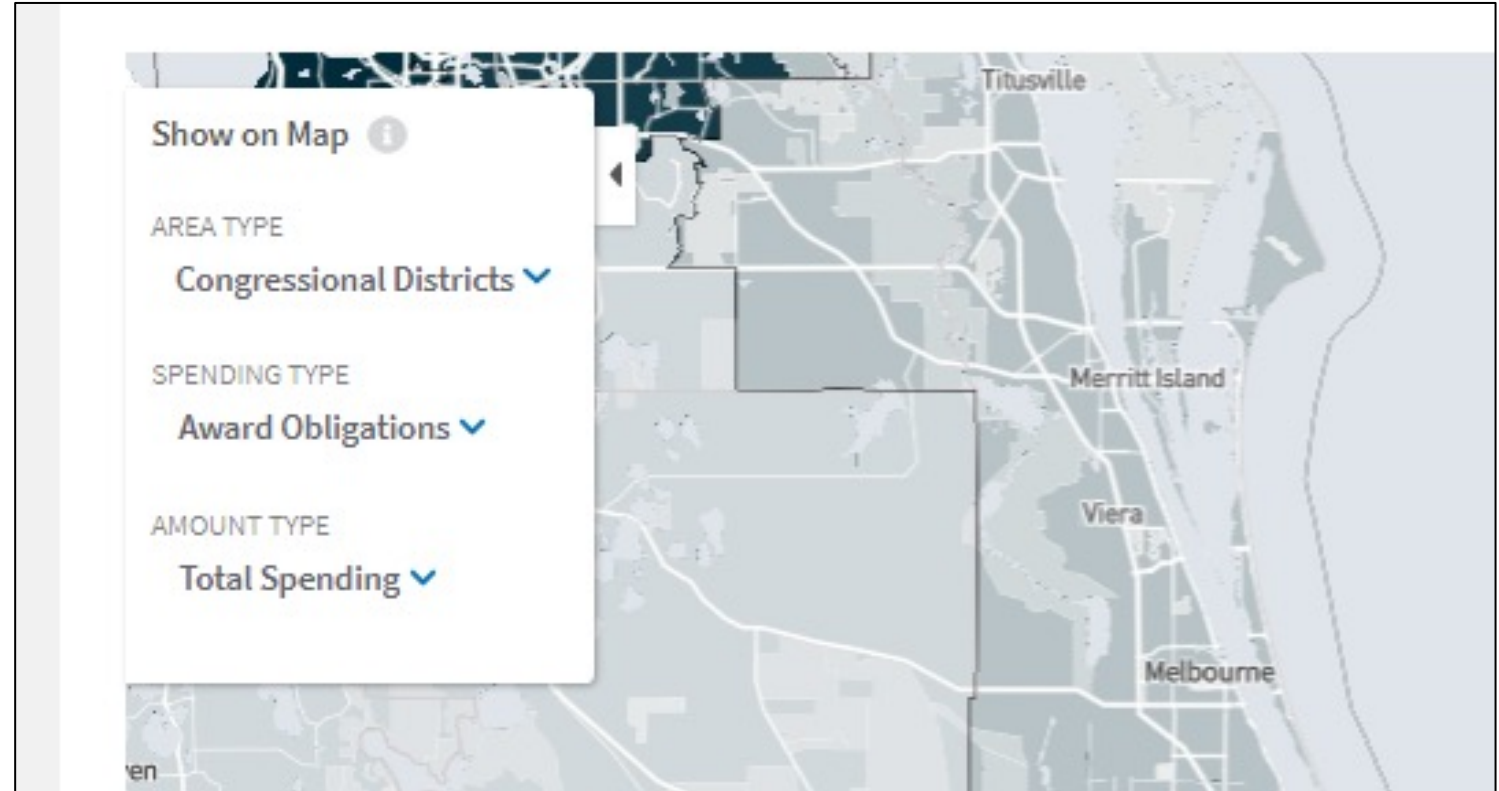


Demo continued

The network inspector will only list API requests since you began inspecting the page.

Each time you update the map (including zooming) a new API request is made.

If you wait to inspect the page until just before you configure the map, it will be easier to find the correct API call.



Demo continued

The screenshot shows the network tab of a web browser's developer tools. The address bar contains `api.usaspending.gov`. The network tab is active, showing a list of requests. The most recent request is selected, and its details are displayed in the right-hand pane. The details pane shows the following information:

- Request URL:** `https://api.usaspending.gov/api/v2/disaster/spending_by_geography/`
- Request Method:** `POST`
- Status Code:** `200` (with a green checkmark icon)

The left-hand pane shows the request name: `spending_by_geography/` and the full URL: `api.usaspending.gov/api/v2/disas...`. The top of the developer tools shows various filters and checkboxes, including `Invert`, `Hide data URLs`, `Blocked Requests`, and `3rd-party requests`.

Select the most recent `spending_by_geography` request and open the Request URL in your browser to study the documentation.

Demo continued

```
{filter: {def_codes: ["L", "M", "N", "O", "P", "U", "V"]},
  filter: {def_codes: ["L", "M", "N", "O", "P", "U", "V"]},
  geo_layer: "district",
  geo_layer_filters: ["1209", "1210", "1206", "1217", "1215"],
  spending_type: "obligation"}
```

- Attributes (object)
 - filter (required, Filter, fixed-type)
 - geo_layer (required, enum[string]) Set the type of shape codes in the response
 - Members
 - state
 - county
 - district
 - geo_layer_filters (optional, array[string]) Allows us to only request data for what is currently in view in the map
 - scope (optional, enum[string]) When fetching awards, use the primary place of performance or recipient location
 - Default: recipient_location
 - Members
 - place_of_performance
 - recipient_location
 - spending_type (required, enum[string])
 - Members
 - obligation
 - outlay
 - face_value_of_loan

Compare the request payload and documentation attributes.

Disaster Spending by Geography

Documentation: https://github.com/fedspendingtransparency/usaspending-api/blob/master/usaspending_api/api_contracts/contracts/v2/disaster/spending_by_geography.md

The `geo_layer_filters` attribute takes a list of 4 digit CD FIPS codes.

The `filter` attribute does not take an `AdvancedFilterObject`, this is different from the Advanced Search endpoints.

The `filter` attribute allows you to filter by DEFC or award type.

- You cannot filter by DEFC using the website alone.
- The API allows you to accurately answer questions about how much an area received by a particular COVID supplemental appropriation bill.

See: <https://www.usaspending.gov/disaster/covid-19/data-sources>

Demo continued

```
1 let
2     url = "https://api.usaspending.gov/api/v2/disaster/spending_by_geography/",
3     body = "{
4         ""spending_type"": ""obligation"",
5         ""geo_layer"": ""district"",
6         ""geo_layer_filters"": [""1208""],
7         ""filter"": {""def_codes"": [""L"", ""M"", ""N"", ""O"", ""P"", ""U"", ""V""]}
8     }",
9     Source = Json.Document(Web.Contents(url,[Content=Text.ToBinary(body),Headers=[#"Content-Type"="application/json"]))),
10    results = Source[results],
11    results1 = results{0},
12    #"Converted to Table" = Record.ToTable(results1)
13 in
14    #"Converted to Table"
```

Use this code in PowerQuery to replicate the API call in Excel.
This code is available for copy/paste in the link below.

Notes: <https://github.com/fedspendingtransparency/usaspending-api/wiki/API-Usage---Power-Query-Example>

Advanced Search Exercise (On your own!)

Use Excel to create an API request to replicate an advanced search in USAspending.

Keep learning!

Study the list of available endpoints: <https://api.usaspending.gov/docs/endpoints>

Learn more about REST APIs: <https://www.redhat.com/en/topics/api/what-is-a-rest-api>

Learn more about API requests in Python: <https://realpython.com/python-requests/>