Tips and Tricks for using CIM

This article describes how to access data in the Colorado Information Marketplace (CIM), the State of Colorado's official open data portal and the data resource for Colorado CDO Analytics Challenge.

Competition participants can explore Colorado public data through the **Colorado Information Marketplace (CIM)** public data portal and learn tips and tricks for using CIM.

Introduction to Colorado Information Marketplace

The Colorado Information Marketplace is provided by the Colorado Governor's Office of Information Technology to make Colorado public data available to citizens. It offers a dataset catalog and dataset repository, an application programming interface (API) and several tools for exploring and visualizing data.

Finding Data

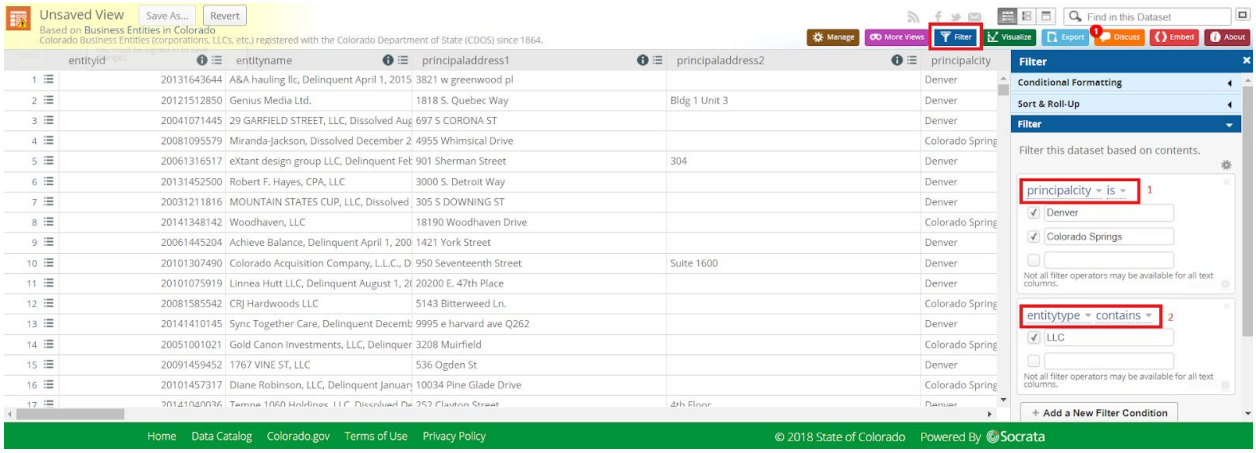
One great feature of CIM is its ability to create filtered views on tables, save them, and make them publicly available to others. When a user is looking for a known dataset, seeing data interpretations that have been completed by others can be greatly beneficial.

Look at **Business Entities** as an example:

- **Colorado Businesses in Good Standing**
- **Denver Businesses in Good Standing**

There is a detriment to the filtered views feature, however, and it comes into play for users who don't know which dataset they're looking for. In this scenario, simply browsing the catalog can seem a bit overwhelming, because users are looking for ONLY the raw data in an effort to see which unique datasets are available to them. A simple solution is to view just "Datasets" or "Maps".

Roll-up and Sort can also be implemented as a method to begin exploring the dataset. The type of data will dictate whether either of these actions will reveal any interesting information. Example:



1. In the View Data screen of any dataset, select the Filter option at the top right. A filter wizard pops up and allows the user to enter filters to sort the dataset.
2. Box 1 is the first filter and can be selected from one of two dropdowns.
    a. The first being the column to sort by
    b. The second being a condition: is, is not, contains, etc.

3. The next step is to enter parameters to match the conditional statement.
    a. In this case, filter where principal city is Denver or Colorado Springs.
4. You may add additional filters (Box 2) by clicking “Add a New Filter Condition.”
5. The final filtered dataset here shows business entities in Denver or Colorado Springs that are Limited Liability Companies.

What is an API Endpoint?

An endpoint in a SODA API is simply a unique URL that represents an object or collection of objects. Every Socrata dataset, and even every data record, has its own endpoint. The endpoint is to point an HTTP client at to interact with data resources.

All resources are accessed through a common base path of /resource/ along with their dataset identifier. This paradigm holds true for most every dataset in every SODA API.
There are two types of dataset identifiers:

Simple Socrata datasets have a unique “4x4” identifier - eight alphanumeric characters split into two four-character phrases by a dash. For example, State Collected City Sales Tax CSV data set:https://data.colorado.gov/Revenue/City-Areas-Collecting-State-Sales-Tax-in-Colorado/wx84-he7r has a 4x4 of “wx84-he7r.”

Note: The 4x4 identifiers are generated by Socrata and are unique within each data site.
The endpoint URL for any SODA API can be created via this simple rule:

https://$domain/resource/$dataset_identifier

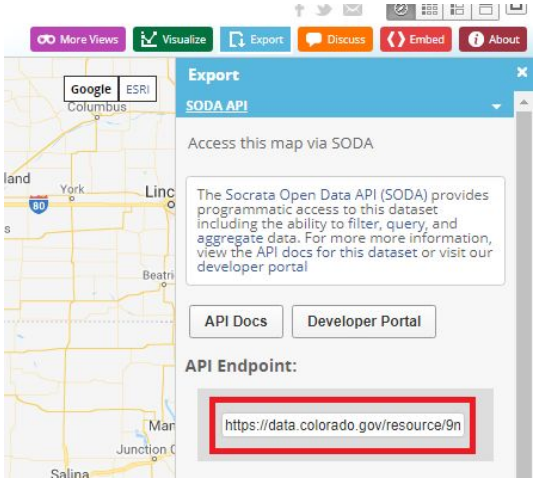
For example, the endpoint for the State Collected City Sales Tax dataset would be:
https://data.colorado.gov/resource/wx84-he7r.json

It is important to note that this is not the case for ALL datasets. As of 2018, geospatial datasets have a different API 4x4 than the main dataset.

For example:
Noxious Weeds in Colorado 2014 has a 4x4 of mguq-rjzb while the API endpoint is https://data.colorado.gov/resource/9mid-gqwm.json. Try to use the dataset 4x4 to discover the endpoint points to a dataset that doesn’t exist.

To find the API endpoint for geospatial datasets, look to the “Developers” section of Socrata-powered data sites or they can be found by clicking the “Export” button on the top right, then clicking “SODA API.” The endpoint location will be under “API Endpoint” field (see photo

below).



Also, find API endpoints in the “Developers” section of Socrata-powered data sites, or under “Export” then “API” on any Socrata dataset page.

Access Methods

There are three primary methods to access data in the CIM.

- Through the CIM web application
- Through the Socrata Open Data API (SODA)
- In Microsoft Excel using an OData link

Access Through the Web Application

The CIM web application enables searching and browsing datasets and views. Browse datasets by view type (e.g., “datasets”, “maps”), by category (e.g., “demographics”), or by category (e.g., “Local Aggregation”). The CIM contains multiple public datasets, but those datasets published specifically for Go Code Colorado will be “tagged” with “gocodecolorado” topic **keywords**.

Use the CIM web application to **view the dataset** itself in one of several views:

- as a table
- as a rich list
- as single records

The CIM web application provides several other features for **exploring** datasets:

- Filter

- Sort
- Visualize (charts, map, etc.)
- View metadata

Export the datasets in several formats:

- CSV
- JSON
- PDF
- RDF
- RSS
- XLS
- XLSX
- XML

And for geospatial datasets:

- KML
- KMZ
- SHP
- Original
    - Note that this is the only option to maintain the original dataset projection - otherwise the projection will export as WGS84
- GeoJSON
- JSON (non-geospatial)
- CSV (non-geospatial)

Then import the datasets into local and or project data architecture/platforms.

Access Data Through the SODA API

Browse CIM by scrolling down to the data catalog and using the search box and browse filters to find datasets of interest. Every dataset is accessible via the SODA API. SODA provides an open, standards-based, RESTful application programming interface to access datasets in the CIM.

- Every hosted dataset is readily and uniformly accessible
- Data can be displayed, and advanced operations such as searching and filtering can be performed
- Datasets can be combined with other Web services to create new mashups and applications
- Socrata Open Data API (SODA) specifications were released to the Open Data community under a Creative Commons license

SODA is supported by a developer site http://beta.dev.socrata.com/.

The Socrata Open Data API allows programmatic access open data resources from from the CIM. Click the link below and for an example on how queries are structured in SODA.

```
http://soda.demo.socrata.com/resource/earthquakes.json?$where=magnitude > 5.5
```

The SODA API documentation has been broken down into three user categories: App Developers; Data Publishers; Data Consumers.

Paging

For performance, SODA APIs are paged and return a maximum of 1000 records per page. So, to request subsequent pages, use the $limit and $offset parameters to request more data. The $limit parameter chooses how many records to return per page, and $offset tells the API on what record to start returning data. Read the detailed documentation on Paging for more info.

Side by Side Comparison of CIM Filter and API Where Query

Filtering data via a SODA API is fairly straightforward. There are two primary mechanisms available to filter data: Simple Filters and SoQL Queries

- *Simple Filters*

  SODA APIs are self-describing - the schema and contents of the dataset itself determines how it can be queried. Any field within the data can be used as a filter simply by appending it to the API endpoint as a GET parameter.

  Additional filters can be added and the filters will be “AND”ed together. Read the detailed documentation on Filtering Datasets for more info.

- *SoQL Queries*

  The ”Socrata Query Language” (SoQL) is a simple, SQL-like query language specifically designed for making it easy to work with data on the web. The language is both powerful and easy to learn and everything works via GET parameters. Many different functions are available via SoQL. Read the detailed documentation on SoQL Queries for more info.

- A great resource for API testing is Postman. It is a free API development tool for sending requests, savings responses, creating tests, creating workflows, and sharing.

Examples using a filter to return a single row of data in the data browser on CIM:
To return a single record by using a filter to select a single record:

Examples using a browser to return JSON data from an API endpoint:
To return a single record by selecting a unique id from a given column:
https://data.colorado.gov/resource/cg8h-amtj.json?$where=location_wdid = “2400583”

Access in Microsoft Excel through an OData Link

The CIM provides an OData link for every hosted dataset. With OData, create a spreadsheet in Microsoft Excel to analyze or visualize a dataset. By using an OData link, the data will be refreshable. Use the OData link with Microsoft Excel 2010, Microsoft Excel 2013 or Microsoft Power Query.

For more details check out the Go Code Colorado Website
http://gocode.colorado.gov/data/colorados-data-repository/ .

The Socrata Open Data API (SODA) allows software developers to access data hosted in Socrata data sites programmatically. Developers can create applications that use the SODA APIs to visualize and “mash-up” Socrata data sets in creative ways.

How to work with API Endpoints Filters, Verbs, Parameters

Working with an API can often be intimidating for those considering themselves average or beginner users. It can even be difficult to visualize what an API could do, when being only familiar with browsing data in a data catalog. So to transcend this gap, it is best to start with simple exercises, and compare the results to the known results of what is returned when using sort, roll up, filter, etc. on the Colorado Information Marketplace data display page.

To view an entire dataset on CIM is the equivalent of performing a GET request on an API endpoint using an HTTP client (such as a web browser, a simple utility like cURL, or a REST client like Postman), automatically get the first 1000 unfiltered records will be displayed, just like the GUI in the Catalog will display the first 1000 unfiltered records.. To narrow down what's returned, use SODA 2.0's filtering capability or filter the data using the GUI in the browser.

Using the $where Parameter

The simplest form of filtering is equality. Use the field name as a query parameter, and its value as the filter value. The request will return only results where that field is equal to that value.
Use the $where query parameter to only return results that meet certain criteria. The $where parameter supports a number of comparison operations:

Operations	Description
<	Less than (numbers) or alphabetically before (strings)
<=	Less than or equal (numbers) or alphabetically before or equal (strings)
=	Equal
!=	Not equal
<	Greater than (numbers) or alphabetically after (strings)
<=	Greater than or equal (numbers) or alphabetically after or equal (strings)
starts_with(x, y)	String x starts with y (strings only)
contains(x,y)	String x contains string y (strings only)
IS NULL	Whether a value is null
IS NOT NULL	Whether a value is not null

In addition, it supports these Boolean operations:

- AND
- OR
- NOT

Parenthesis can be used as to specify the order of operations.
For more details, see the queries page.

Examples Using the API Where Query

The following example returns only earthquakes with a magnitude greater than 5:

http://soda.demo.socrata.com/resource/earthquakes.json?$where=magnitude > 5

The following example returns only earthquakes with a magnitude greater than or equal to 5 but less than 5.5:
http://soda.demo.socrata.com/resource/earthquakes.json?$where=magnitude >= 5 AND magnitude < 5.5

Number and Boolean Operations with Get

Number

Numbers are arbitrary precision, arbitrary scale numbers. They can represent any number exactly, except for some numbers that repeat infinitely. Since, numbers can be larger than doubles allow and more precise, many formats need to serialize them as strings.

Operation	Description
<	True for numbers less than this one.
<=	True for numbers that are less than or equal to this one.
=	True for numbers that are equal to this one.
!=	True for numbers that are not equal to this one.
>	True for numbers that are greater than this one.
>=	True for numbers that are greater than or equal to this one.
*	Multiplies two numbers
/	Divides one number by another
+	Adds two numbers
-	subtracts one number from another

Double

A double (used to be number) is an IEEE floating point double. They are easy to use and are represented as numbers in all the response formats, however, they have less precision than the regular Number data type. Currently, all numbers are stored as Numbers. Doubles exist as a way of making responses easier to consume, in the case the caller wants results that are easier to parse and can potentially give up precision to do so.

Operation	Description
<	True for numbers less than this one.
<=	True for numbers that are less than or equal to this one.
=	True for numbers that are equal to this one.

!=	True for numbers that are not equal to this one.
>	True for numbers that are greater than this one.
>=	True for numbers that are greater than or equal to this one.
*	Multiplies two numbers
/	Divides one number by another
+	Adds two numbers
-	subtracts one number from another

Boolean

Booleans can be either true or false.

Operator	Description
AND	The logical **and** of two expressions.
OR	The logical **or** of two expressions.
NOT	The logical **not** of an expression.

Throttling and Applications Tokens

A certain number of requests can be made without an application token, but they come from a shared pool and an individual connection will eventually going to get cut off.

To get more requests, register for an application token and to be granted up to 1000 requests per rolling hour period. Special exceptions for projects requiring more than 1000 requests per rolling hour period can be provided upon request. Use the Help! tab on the right of this page to file a trouble ticket, and read the detailed documentation on Paging for more info.

Differences between SODA 2.0 and 2.1

When working with the SODA end points it is important to understand some of the differences between them. SODA 2.0 was originally released in 2011. Although 2.1 is backwards-compatible with 2.0, there are a number of differences between the two APIs:

- 2.0 supports fewer SoQL functions than 2.1.
- The only geospatial data type supported is the Location datatype
- Text comparisons are case-insensitive

Also important to note is tabular datasets are created and initially stored in what is call the Original Back End (OBE). Upon creation, a copy of the dataset is replicated to the New Back End (NBE). Each copy of the dataset will have a different UID. This means that 2.1 endpoint

will be different that the 2.0 endpoint. The other major difference is when data is updated in the OBE it can take up to 1 hour before the NBE (2.1) endpoint is showing that new information.

Additional Detailed References:

- https://dev.socrata.com/docs/functions/
- https://dev.socrata.com/docs/endpoints.html
- https://support.socrata.com/hc/en-us/articles/115012661308-Socrata-Data-Refresh-Process

CIM API Example with CDHE Degrees Awarded

https://dev.socrata.com/foundry/data.colorado.gov/yt5k-hawq

API: CDHE - Degrees Awarded

The Degree File includes all students who have received a certificate, degree, or formal award approved by DHE during the report year. Degrees earned but not conferred during the report period should be included in the following year's report. The file is collected annually for federal and state reporting. A summer degree file is collected for use in graduation rate calculations, but these records are reported again in the full year file.

"Get" view exposes:

Parameter	Value	Type	Description

All communication with the API is done through HTTPS, and errors are communicated through HTTP response codes. Available response types include JSON, XML, and CSV, which are selectable by the "extension" on API requests or by HTTPAccepts headers.

This documentation also includes inline, runable examples. Click on any link that contains a gear symbol next to it to run that example live against the API: CDHE - Degrees Awarded API.

Fields
Below is an example of the fields are available on the API: CDHE - Degrees Awarded API.

Field Name: year
Human Name: year
Type: number

Examples

Copy and paste these to a browser as examples

Retrieve all records with `year` equal to `2012`:

GET https://data.colorado.gov/resource/cdhe-degrees-awarded.json?year=2012

Retrieve all records that are greater than 2001:

GET https://data.colorado.gov/resource/cdhe-degrees-awarded.json?$where=year > 2001

View more detailed documentation for the number datatype »