

# MANUAL DATOS ABIERTOS EN COLOMBIA

**PARA:**

**DESARROLLADOR**

En el siguiente documento se explica el procedimiento para realizar las diferentes acciones que posibilita la herramienta Socrata



**MINTIC**

**ASESOFTWARE**  
25 AÑOS



**vive digital**  
para la gente



**TODOS POR UN  
NUEVO PAÍS**  
PAZ EQUIDAD EDUCACIÓN

# TABLA DE CONTENIDOS

1.	PROPÓSITO DEL DOCUMENTO.....	1
2.	FRONT-END DEL PORTAL DE DATOS ABIERTOS.....	2
2.1.	INTERFAZ DE LA PÁGINA DE INICIO.....	2
2.2.	INTERFAZ DEL CATÁLOGO DE DATOS.....	4
2.3.	INTERFAZ DEL CONJUNTO DE DATOS.....	5
2.3.1.	OPCIONES DEL PANEL DE EXPORTAR.....	6
3.	QUERY API.....	10
3.1.	FILTRO SIMPLE.....	10
3.2.	LENGUAJE DE CONSULTA SOCRATA.....	11
3.2.1.	SELECT.....	12
3.2.2.	WHERE.....	14
3.2.3.	ORDER.....	15
3.2.4.	GROUP.....	16
3.2.5.	HAVING.....	17
3.2.6.	LIMIT.....	17
3.2.7.	OFFSET.....	18
3.2.8.	BUSQUEDAS.....	19
3.2.9.	QUERY.....	20
3.3.	BÚSQUEDAS GEOGRÁFICAS.....	21
3.3.1.	WITHIN_BOX (... ).....	21
3.3.2.	WITHIN_CIRCLE (... ).....	23
3.3.3.	WITHIN_POLYGON (... ).....	25
4.	DATASYNC.....	27
4.1.	TOKEN PARA APLICACIONES.....	28

4.2.	STANDARD JOB .....	28
4.2.1.	MAPEADO DE COLUMNAS SINTÉTICAS .....	31
4.2.2.	OPCIONES DE IMPORTACIÓN AVANZADA .....	32
4.3.	PORT JOB.....	33
4.4.	METADATA JOB .....	34
4.5.	CONJUNTOS DE DATOS EN FORMATO XML.....	34
4.5.1.	UTILIZANDO SQL SERVER INTEGRATION SERVICES .....	36
4.5.2.	UTILIZANDO SODA API.....	43
5.	SOPORTE DE SOCRATA .....	46
5.1	PÁGINA DE SOPORTE .....	46
5.2	GENERAR UN TOKEN.....	49

# 01

# PROPÓSITO DEL DOCUMENTO

En el presente documento se pretende recoger información de utilidad, a modo de tutorial, para desarrolladores quienes son usuarios con rol en Socrata desde espectadores hasta administradores, enfocados en desarrollos de APIs, ETL, o programas en JAVA, consumiendo datos en Socrata.

Este documento esta categorizado en 2 áreas principales:

- Front-end del portal de datos abiertos: Sitio web utilizado para visualizar los datos principalmente. En este apartado nos centraremos en las posibilidades que ofrece su interfaz y en la gestión de los datos.
- DataSync: Una aplicación java creada por Socrata pensada para ser utilizada a modo de herramienta ETL (Extracción, Transformación y Carga) para subir datos de manera automática al portal de datos abiertos del estado colombiano, entre otras operaciones.

Para profundizar en cualquier concepto o funcionalidad se puede consultar la documentación oficial de Socrata disponible en <https://support.socrata.com/> donde también se pueden registrar “tickets” con dudas o preguntas dirigidas al servicio técnico de Socrata.

02

# FRONT-END DEL PORTAL DE DATOS ABIERTOS

## 2.1. INTERFAZ DE LA PÁGINA DE INICIO

La página de inicio o, en inglés, “Home Page” tiene una apariencia por defecto, diseñada para Colombia al momento de la contratación del portal. A continuación, dicha página de inicio:



 <p>Datos del Gobierno Colombiano para investigar, desarrollar aplicaciones, crear visualizaciones e historias</p> <p><b>87</b> datos</p>	 <p>Datos de Gobierno para que puedan ser usados en investigaciones, aplicaciones, visualizaciones e historias</p> <p><b>6</b> entidades</p>	 <p>Visualizaciones e investigaciones creadas a partir de datos abiertos</p> <p><b>11</b> visualizaciones</p>			
<h2>NOVEDADES</h2>					
					
<p><b>Open Data Barometer</b></p> <p>Colombia ascendió 12 puestos en el Open Data Barometer 2015, alcanzando el lugar 28 en este índice que mide el impacto de las iniciativas de datos abiertos en el mundo. El país ocupa el cuarto lugar en Latinoamérica después de México (16), Brasil (17) y Uruguay (19)</p>	<p><b>Periodismo con datos</b></p> <p>Gracias a un conjunto de datos publicado en el portal de datos abiertos, el periódico El Tiempo publicó una investigación donde se toma la mejor y la peor agua del país. Los datos se desprenden del Índice de Riesgo de la Calidad del Agua para Consumo Humano</p>	<p><b>Tour de estudio Panamá</b></p> <p>Durante cuatro días una delegación de la República de Panamá estuvo en el Ministerio de las TIC con el fin de fortalecer sus capacidades para proporcionar acceso a la información para el desarrollo sostenible a través de los Datos Abiertos de Gobierno. El tour estuvo liderado por UNDESA.</p>	<p><b>Jueves de Gobierno Abierto</b></p> <p>Representantes de más 120 entidades estuvieron presentes en la capacitación sobre la nueva plataforma de Datos Abiertos. En el marco de la jornada se realizaron ejercicios de visualización, cargue de conjuntos de datos y visualizaciones análogas</p>		
<p>Políticas de Privacidad y Condiciones de uso   Visitas: 29794          Ministerio de Tecnologías de la Información y las Comunicaciones          Edificio Murillo Toro Cra. 8a entre calles 12 y 13, Bogotá, Colombia - Código Postal 111711          Línea Anticorrupción: 01-80009 10 742, Bogotá 595 3595 - Correo: sportecc@mintic.gov.co          Horario: Lunes a Viernes 8:30 am - 4:30 pm.</p>					

La página de inicio contiene los logos correspondientes, un panel de enlaces, un buscador que accede al catálogo de datos, un mensaje de bienvenida, una sección con accesos rápidos a ciertas partes del portal conocida como área de *Novedades* y finalmente una barra de enlaces propios de Socrata.

## 2.2. INTERFAZ DEL CATÁLOGO DE DATOS

Una vez entramos al catálogo de datos vemos la siguiente pantalla:

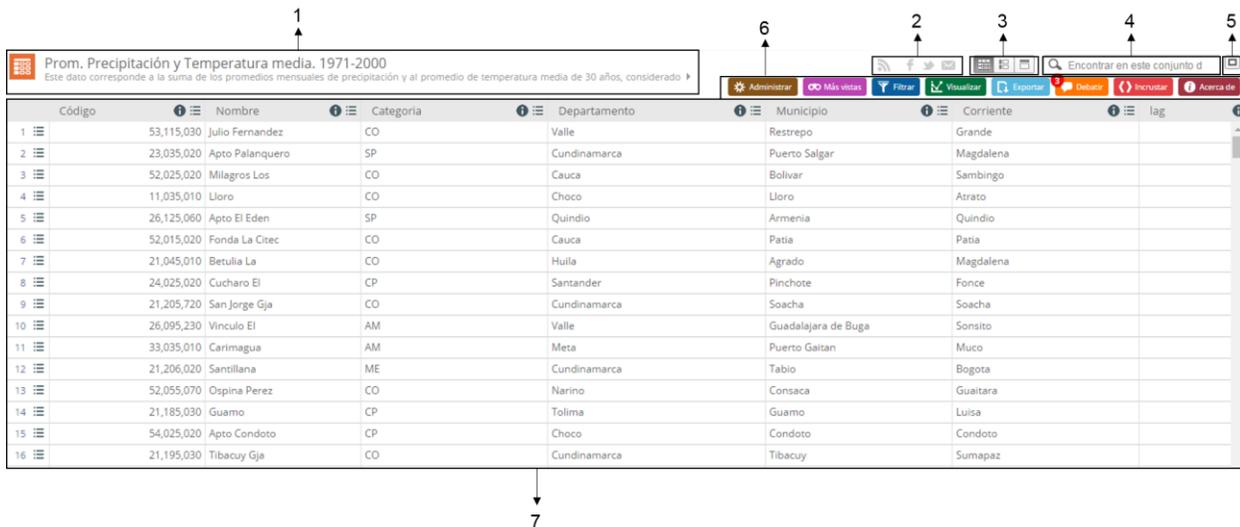
The screenshot shows a web interface for a data catalog. At the top, there is a search bar labeled 'Búsqueda'. Below it, the text '425 Resultados' is displayed. To the right, there is a dropdown menu for 'Ordenar por' set to 'Más relevante'. On the left side, there are three filter sections: 'Categorías' with options like 'Agricultura y Desarrollo Rural', 'Ambiente y Desarrollo Sostenible', 'Comercio, Industria y Turismo', 'Cultura', and 'Economía y Finanzas'; 'Tipos de vista' with options like 'Archivos y documentos', 'Calendarios', 'Conjuntos de datos', 'Conjuntos de datos externos', and 'Data Lens pages'; and 'Palabras clave' with options like 'colombiatic, internet, banda ancha', 'catálogo estaciones', 'clima', and 'fuentes de información para innovaciones'. The main content area displays a list of data sets. Each entry includes a title, a description of topics, the date it was last updated, and the number of views. The entries shown are: 'Novedades' (no topics assigned, updated June 20, 2016, 308 views), 'Documentos iniciativa Guía de Datos Abiertos, Manuales' (manuals for open data, updated June 28, 2016, 133 views), 'Preguntas Frecuentes - FAQ' (dataset with frequent questions, updated June 22, 2016, 131 views), 'Prom. Precipitación y Temperatura media. 1971-2000' (dataset of monthly precipitation and temperature averages, updated June 10, 2016, 114 views), and 'Dataset Fact Bar' (no topics assigned, updated June 10, 2016).

Como se puede ver en la imagen, los conjuntos de datos y las vistas aparecen listados y paginados cada 10 elementos. Se nos permite ordenarlos y filtrarlos por tipología (si se trata de conjuntos de datos tabulares, gráficos, mapas...), por categorías o por palabras clave. Así mismo, dispone de un buscador asociado al catálogo que busca coincidencias por títulos de conjuntos de datos, descripción del conjunto de datos, vistas o por palabras clave.

Si clicamos sobre un conjunto de datos, se despliega la información de dicho conjunto de datos.

## 2.3. INTERFAZ DEL CONJUNTO DE DATOS

La interfaz de los conjuntos de datos tiene la siguiente apariencia:



1

6

2

3

4

5

Código	Nombre	Categoría	Departamento	Municipio	Corriente	lag
1	53,115,030 Julio Fernandez	CO	Valle	Restrepo	Grande	
2	23,035,020 Apro Palanquero	SP	Cundinamarca	Puerto Salgar	Magdalena	
3	52,025,020 Milagros Los	CO	Cauca	Bolivar	Sambingo	
4	11,035,010 Lloro	CO	Choco	Lloro	Atrato	
6	26,125,060 Apro El Eden	SP	Quindio	Armenia	Quindio	
6	52,015,020 Fonda La Citec	CO	Cauca	Patia	Patia	
7	21,045,010 Betulia La	CO	Huila	Agrado	Magdalena	
8	24,025,020 Cucharo El	CP	Santander	Pinchote	Fonce	
9	21,205,720 San Jorge Gja	CO	Cundinamarca	Soacha	Soacha	
10	26,095,230 Vinculo El	AM	Valle	Guadalajara de Buga	Sonsito	
11	33,035,010 Carimagua	AM	Meta	Puerto Gaitan	Muco	
12	21,206,020 Santillana	ME	Cundinamarca	Tabio	Bogota	
13	52,055,070 Ospina Perez	CO	Narino	Consaca	Guaicara	
14	21,185,030 Guamo	CP	Tolima	Guamo	Luisa	
15	54,025,020 Apro Condoto	CP	Choco	Condoto	Condoto	
16	21,195,030 Tibacuy Gja	CO	Cundinamarca	Tibacuy	Sumapaz	

7

En esta interfaz aparecen los siguientes elementos:

- 1) Icono, título y descripción

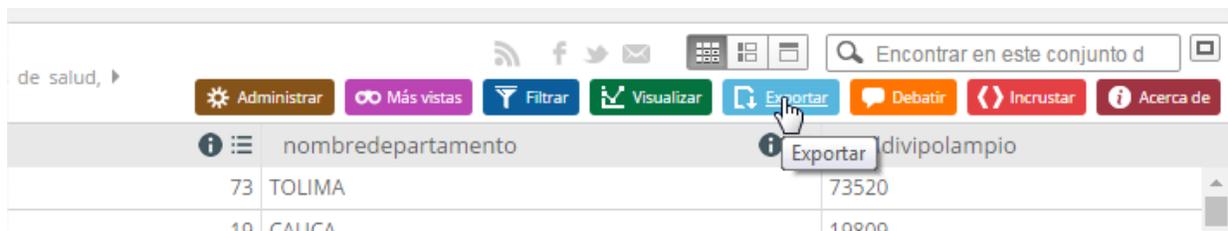
*Nota: El icono se puede modificar para cada conjunto de datos, el que aparece por defecto corresponde al icono de la tipología de datos.*

- 2) Enlaces de suscripción y compartición: El primer icono nos permite suscribirnos al conjunto de datos en concreto vía RSS o vía correo electrónico para ser informados de cualquier cambio en él y el resto de iconos nos permiten compartir el conjunto de datos mediante las redes sociales y correo electrónico.
- 3) Opciones de visualización: Estos tres botones nos permiten tener distintas formas de visualizar el conjunto de datos pudiendo seleccionar más de uno al mismo tiempo. El primero es el que está activado por defecto y representa los datos en forma de tabla, el segundo los muestra como una lista completa con paginación y el último, muestra los datos a nivel de fila o registro, también con la pertinente paginación.
- 4) Panel de búsqueda: Permite buscar palabras o cifras dentro del conjunto de datos.
- 5) Botón de pantalla completa: Este botón permite visualizar el conjunto de datos a pantalla completa con lo que se elimina la cabecera y los enlaces de la parte inferior.

- 6) Barra de menús: Esta barra contiene las pestañas de menú con los que podemos interactuar con los datos. Se muestran botones de: Editar, Administrar, Mas vistas, Filtrar, Visualizar, Exportar, Debatir, Incrustar, Acerca de. Las funciones que permiten estos menús se explican en detalle en el Tutorial de Socrata para el Usuario Final.
- 7) Datos: En este apartado se muestra el conjunto de datos, la vista, el enlace externo, el documento, etc. que corresponda.

## 2.3.1. OPCIONES DEL PANEL DE EXPORTAR

La opción exportar en la barra de menú del conjunto de datos, cuenta con opciones para los desarrolladores, estas opciones también se encuentran en el Tutorial de Socrata para el Usuario Final, pero aquí nos enfocaremos en su uso para desarrolladores.



### 2.3.1.1. SODA API

Esta sección del panel exportar nos facilita la información necesaria para usar la API SODA con ese conjunto de datos en cuestión. La API que provee Socrata está basado en lenguaje JAVA y se puede descargar de la siguiente web <https://github.com/socrata/soda-java> totalmente libre.

socrata / soda-java

Watch 62 Star 37 Fork 25

Code Issues 1 Pull requests 0 Pulse Graphs

This is the Java API for the SODA 2.0 API <http://socrata.github.com/soda-java>

148 commits 9 branches 18 releases 14 contributors

Branch: master New pull request Find file Clone or download

chitang committed on GitHub Truncate should not be delete. (#16) Latest commit e457623 20 days ago

project	Add support for building an assembly with all dependencies to make it...	8 months ago
src	Truncate should not be delete. (#16)	20 days ago
.gitignore	Build with SBT instead of Maven	2 years ago
README.md	Correct spelling mistake in README.md	4 months ago
assembly.sbt	Add support for building an assembly with all dependencies to make it...	8 months ago
build.sbt	Integrate various dependency versions	8 months ago

En esta web JavaDoc se explica cada uno de los componentes que se encuentran en la API. Así mismo, se puede usar la siguiente URL <http://socrata.github.io/soda-java/> para consultar ejemplos prácticos usando la API de Java.

## Consumer

### Simple Query

The consumer API is simple. The following example will issue two requests, one will return the results from the "test-data" dataset, as the JSON string. The other will return the results as the `Nomination` java objects:

```
Soda2Consumer consumer = Soda2Consumer.newConsumer("https://sandbox.demo.so

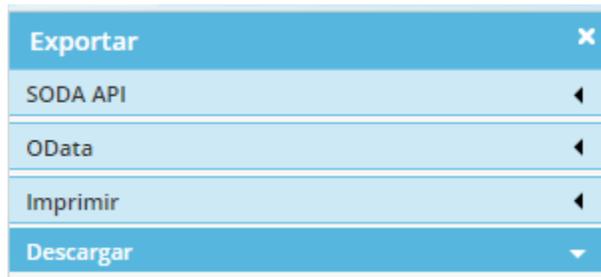
//To get a raw String of the results
ClientResponse response = consumer.getHttpLowLevel().query("nominationsCopy
String payload = response.getEntity(String.class);
System.out.println(payload);

//Get get this automatcally serialized into a set of Java Beans annotated w
List<Nomination> nominations = consumer.query("nominationsCopy", SoqlQuery.
TestCase.assertTrue(nominations.size() > 0);
System.out.println(nominations.size());
```

### Building Queries

Along with the consumer API, is a builder class to make it easier to build the SoQL queries. For example, to query for the name, position and nomination date of nominees for the Department of State, sorted by position:

Una vez se tenga implementada la API de Java, y se quiera hacer uso de un conjunto de datos en específico desde Socrata, mediante la opción exportar, SODA API, como sigue:



Exportar

Documentación de la API

Portal del desarrollador

Parámetro de acceso de la API:

Identificaciones de columna

Ciudad	ciutat
Número de Habitante	nombre_d_habitants
Area (Km)	rea_km
Densidad de Habitante	densitat_d_habitants_hab.
Porcentaje de la pobla	percentatge_de_la_poblac
Ubicación Ayuntamier	ubicaci_ajuntament

- El botón de Documentación de la API nos dirige a un sitio web con información “a medida” para ese conjunto de datos. En ella encontraremos ejemplos de uso, los identificadores de columna, instrucciones de cómo realizar operaciones sobre cada columna... Esta funcionalidad se llama “Fundición API” (API Foundry).
- El botón de Portal del Desarrollador nos dirige al portal de desarrolladores de Socrata donde podemos consultar guías del uso de la API, descargar las librerías y SDKs para diversos lenguajes de programación...
- El parámetro de acceso directo a la API es el recurso que debemos introducir para acceder a este conjunto de datos en concreto mediante la API.
- Los identificadores de columna son los “códigos” que debemos usar para referenciar cada columna.

Podemos hacer uso de este SODA API, de varias maneras, pero la más sencilla es utilizarlo a través del Endpoint REST, para ver lo anterior a más detalle se puede consultar el siguiente link que contiene un video tutorial paso a paso, <https://www.youtube.com/watch?v=awB0vxXVNVc>

# 03

# QUERY API

## 3.1. FILTRO SIMPLE

Es posible crear consultas de bases de datos con simples filtros de igualdad, para ellos es primordial haber iniciado sesión en [www.datos.gov.co](http://www.datos.gov.co), sólo tiene que utilizar el nombre del campo de la columna como su parámetro y el contenido que desea filtrar como su valor.

Por ejemplo, para un conjunto de datos que contiene un listado de notarías.

```
https://www.datos.gov.co/resource/7hb3-hfs7.json

[{"direcci_n": "Avenida Calle 134 No. 78-83 LC 1", "localizaci_n_soocrata": {"type": "Point", "coordinates": [-74.029457, 4.711811]}, "nombre_notario": "MARIA IN\u00c9S PANTOJA PONCE", "notaria": "69", "telefono": "6266069-6152969"}, {"direcci_n": "Calle 128 No.8a-34 LC 9 - 10 y 11", "localizaci_n_soocrata": {"type": "Point", "coordinates": [-74.075382, 4.600616]}, "nombre_notario": "MANUEL ) CAROPRESE M\u00c9NDEZ", "notaria": "3", "telefono": "4789177 - 4789179"}, {"direcci_n": "Carrera 8 No.17-30 Piso 3, 4", "localizaci_n_soocrata": {"type": "Point", "coordinates": [-74.073193, 4.604102]}, "nombre_notario": "LINA MARIA RODRIGUEZ", "notaria": "4", "telefono": "6051313-2860629"}, {"direcci_n": "CRA 15 A No 120 - 63", "localizaci_n_soocrata": {"type": "Point", "coordinates": [-74.04326, 4.69937]}, "nombre_notario": "ANDR\u00c9S HIBER AR\u00c9VALO", "notaria": "5", "telefono": "7450597-312909944-3125228217"}, {"direcci_n": "CRA 9 No 69 - 31", "localizaci_n_soocrata": {"type": "Point", "coordinates": [-74.059074, 4.652759]}, "nombre_notario": "MARIA AMPARO QUINTERO ARTURO", "notaria": "6", "telefono": "3170100 - 3145641"}, {"direcci_n": "Calle 12 B No. 8-39 ", "localizaci_n_soocrata": {"type": "Point", "coordinates": [-74.070811, 4.608145]}, "nombre_notario": "LIGIA JOSEFINA ERAZO CABRERA", "notaria": "7", "telefono": "2826565 - 2826563"}, {"direcci_n": "CRA 16 No 80- 90", "localizaci_n_soocrata": {"type": "Point", "coordinates": [-74.057636, 4.666832]}, "nombre_notario": "FABIO CASTIBLANCO CALIXTO", "notaria": "8", "telefono": "6102090"}]
```

Filtraremos las filas para las cuales el campo notaria es igual a 6.

<https://www.datos.gov.co/resource/7hnu-p99i.json?notaria=6>

```
https://www.datos.gov.co/resource/7hnu-p99i.json?notaria=6

[ {
  "nombre_notario" : "MARIA AMPARO QUINTERO ARTURO",
  "direcci_n" : "CRA 9 No 69 - 31",
  "notaria" : "6",
  "telefono" : "3170100 - 3145641",
  "localizaci_n_soocrata" : {
    "latitude" : "4.652759",
    "needs_recoding" : false,
    "longititude" : "-74.059074"
  }
} ]
```

Si incluye parámetros de filtro adicionales, los filtros se combinarán utilizando un valor lógico AND. Por ejemplo, los siguientes filtros dirección “CRA 13 No 35 - 69 LOCAL 102” y notaria 38. <https://www.datos.gov.co/resource/7hnu-p99i.json?direccion=CRA%2013%20No%2035%20-%2069%20LOCAL%20102&notaria=38>

```
[ {
  "nombre_notario" : "EDUARDO DURAN GÓMEZ",
  "dirección" : "CRA 13 No 35 - 69 LOCAL 102",
  "notaria" : "38",
  "telefono" : "2458721 EXT 102-3",
  "localización_socrata" : {
    "latitude" : "4.622924",
    "needs_recoding" : false,
    "longitude" : "-74.067893"
  }
}
```

## 3.2. LENGUAJE DE CONSULTA SOCRATA

El API de Socrata provee una funcionalidad de consulta a través de queries mediante un lenguaje llamado “Lenguaje de Consulta Socrata” o SoQL por sus siglas en ingles. Como su nombre lo indica está inspirado en el lenguaje SQL, el cual es utilizado por los sistemas de bases de datos relacionales.

Las declaraciones SoQL, se dividen en “parámetros” similares a las cláusulas de SQL. Cada cláusula puede ser expresada ya sea directamente como un parámetro de URL o como una declaración SoQL. Si no se especifica un parámetro, se utiliza el valor predeterminado.

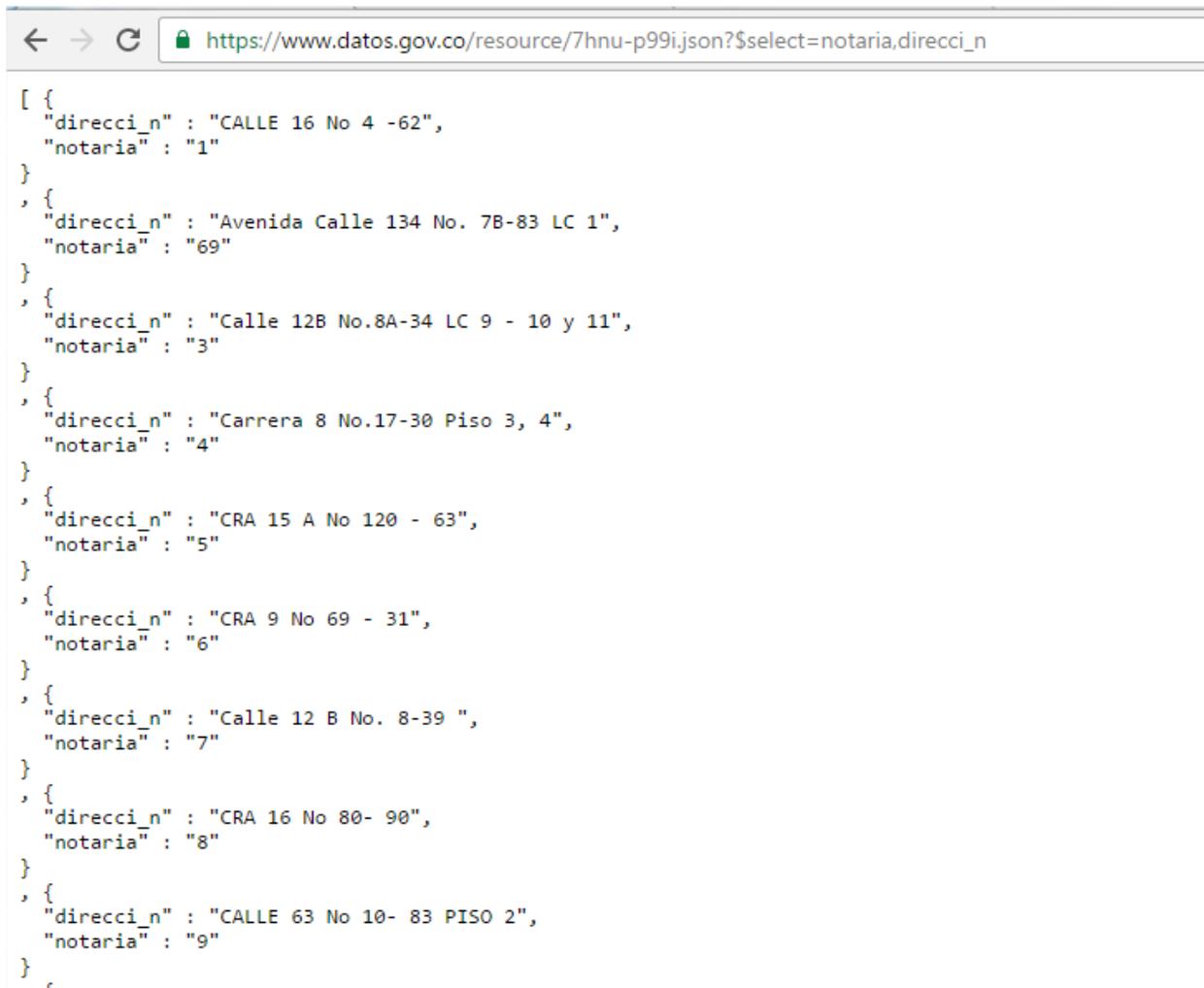
A continuación, se presentan las clausulas más utilizadas por los desarrolladores, pero Socrata expone muchas más las cuales se describen en el siguiente link <https://dev.socrata.com/docs/functions/#>

## 3.2.1.SELECT

El \$select es similar a un SELECT en SQL. Permite la utilización de expresiones aliadas tal como \$group.

Por ejemplo, podemos extraer del conjunto de datos abiertos únicamente las columnas que contienen la dirección y el código de notaria, únicamente debemos separarlas por una coma.

[https://www.datos.gov.co/resource/7hnu-p99i.json?\\$select=notaria,direcci\\_n](https://www.datos.gov.co/resource/7hnu-p99i.json?$select=notaria,direcci_n)



```
[ {
  "direcci_n" : "CALLE 16 No 4 -62",
  "notaria" : "1"
}, {
  "direcci_n" : "Avenida Calle 134 No. 7B-83 LC 1",
  "notaria" : "69"
}, {
  "direcci_n" : "Calle 12B No.8A-34 LC 9 - 10 y 11",
  "notaria" : "3"
}, {
  "direcci_n" : "Carrera 8 No.17-30 Piso 3, 4",
  "notaria" : "4"
}, {
  "direcci_n" : "CRA 15 A No 120 - 63",
  "notaria" : "5"
}, {
  "direcci_n" : "CRA 9 No 69 - 31",
  "notaria" : "6"
}, {
  "direcci_n" : "Calle 12 B No. 8-39 ",
  "notaria" : "7"
}, {
  "direcci_n" : "CRA 16 No 80- 90",
  "notaria" : "8"
}, {
  "direcci_n" : "CALLE 63 No 10- 83 PISO 2",
  "notaria" : "9"
}
]
```

De igual manera podemos crear “alias” a las columnas tal como lo haríamos en SQL. Por ejemplo, colocaremos un alias a la primera columna del select dirección y la nombraremos “alias”.

[https://www.datos.gov.co/resource/7hnu-p99i.json?\\$select=notaria,direcci\\_n%20AS%20alias](https://www.datos.gov.co/resource/7hnu-p99i.json?$select=notaria,direcci_n%20AS%20alias)

```
← → ↻ https://www.datos.gov.co/resource/7hnu-p99i.json?$select=notaria,direcci_n%20AS%20alias

[ {
  "alias" : "CALLE 16 No 4 -62",
  "notaria" : "1"
},
{
  "alias" : "Avenida Calle 134 No. 7B-83 LC 1",
  "notaria" : "69"
},
{
  "alias" : "Calle 12B No.8A-34 LC 9 - 10 y 11",
  "notaria" : "3"
},
{
  "alias" : "Carrera 8 No.17-30 Piso 3, 4",
  "notaria" : "4"
},
{
  "alias" : "CRA 15 A No 120 - 63",
  "notaria" : "5"
},
{
  "alias" : "CRA 9 No 69 - 31",
  "notaria" : "6"
},
{
  "alias" : "Calle 12 B No. 8-39 ",
  "notaria" : "7"
},
{
  "alias" : "CRA 16 No 80- 90",
```

Por ultimo puede modificar la salida del query utilizando operadores. Por ejemplo, listar las direcciones, y las notarías donde el código de la notaria sea multiplicado por 2.13.

[https://www.datos.gov.co/resource/7hnu-p99i.json?\\$select=direcci\\_n,notaria%20\\*%202.13](https://www.datos.gov.co/resource/7hnu-p99i.json?$select=direcci_n,notaria%20*%202.13)

```
← → ↻ https://www.datos.gov.co/resource/7hnu-p99i.json?$select=direcci_n,notaria%20*%202.13

[ {
  "notaria_2_13" : "2.13",
  "direcci_n" : "CALLE 16 No 4 -62"
},
{
  "notaria_2_13" : "146.97",
  "direcci_n" : "Avenida Calle 134 No. 7B-83 LC 1"
},
{
  "notaria_2_13" : "6.39",
  "direcci_n" : "Calle 12B No.8A-34 LC 9 - 10 y 11"
},
{
  "notaria_2_13" : "8.52",
  "direcci_n" : "Carrera 8 No.17-30 Piso 3, 4"
},
{
  "notaria_2_13" : "10.65",
  "direcci_n" : "CRA 15 A No 120 - 63"
},
{
  "notaria_2_13" : "12.78",
  "direcci_n" : "CRA 9 No 69 - 31"
}
```

## 3.2.2.WHERE

El parámetro \$where permite filtrar los resultados de una consulta utilizando operadores lógicos. Por ejemplo, todos los códigos de notarías que sean menores de 10.

[https://www.datos.gov.co/resource/7hnu-p99i.json?\\$where=notaria%20%3C%2010](https://www.datos.gov.co/resource/7hnu-p99i.json?$where=notaria%20%3C%2010)

```
← → ↻ https://www.datos.gov.co/resource/7hnu-p99i.json?$where=notaria%20%3C%2010

[ {
  "nombre_notario" : "HERMANN PIESCHACON F.",
  "direcci_n" : "CALLE 16 No 4 -62",
  "notaria" : "1",
  "telefono" : "3423049",
  "localizaci_n_soocrata" : {
    "latitude" : "4.600958",
    "needs_recoding" : false,
    "longitude" : "-74.07001700000002"
  }
}, {
  "nombre_notario" : "MANUEL J CAROPRESE MÉNDEZ",
  "direcci_n" : "Calle 12B No.8A-34 LC 9 - 10 y 11",
  "notaria" : "3",
  "telefono" : "4789177 - 4789179",
  "localizaci_n_soocrata" : {
    "latitude" : "4.600616",
    "needs_recoding" : false,
    "longitude" : "-74.075382"
  }
}
]
```

También, es posible combinar varios filtros usando los operadores lógicos, para el ejemplo usaremos el operador lógico AND.

[https://www.datos.gov.co/resource/7hnu-p99i.json?\\$where=notaria%20%3C%2010%20AND%20telefono%20=%20%272358799%27](https://www.datos.gov.co/resource/7hnu-p99i.json?$where=notaria%20%3C%2010%20AND%20telefono%20=%20%272358799%27)

```
← → ↻ https://www.datos.gov.co/resource/7hnu-p99i.json?$where=notaria%20%3C%2010%20AND%20telefono%20=%20%272358799%27

[ {
  "nombre_notario" : "LEOVEDIS LEIAS MARTÍNEZ",
  "direcci_n" : "CRA 13 No 64 - 29",
  "notaria" : "2",
  "telefono" : "2358799",
  "localizaci_n_soocrata" : {
    "latitude" : "4.6510923",
    "needs_recoding" : false,
    "longitude" : "-74.0632481"
  }
}
]
```

Existen más operadores lógicos habilitados para esta función, los cuales se listan a continuación:

OPERADOR	DESCRIPCIÓN	EJEMPLO
<b>AND</b>	La lógica Y de dos expresiones.	<b>a AND b</b> retorna VERDADERO si ambas a,b son verdaderas.
<b>OR</b>	La lógica O de dos expresiones.	<b>a OR b</b> retorna VERDADERO si alguna a o b es verdadera.
<b>NOT</b>	La lógica NO de una expresión.	<b>NOT a</b> retorna VERDADERO únicamente cuando a es falso.
<b>IS NULL</b>	Si un valor es nulo o no.	<b>a IS NULL</b> retorna VERDADERO únicamente si a es nulo.
<b>IS NOT NULL</b>	Si un valor no es nulo.	<b>a IS NOT NULL</b> retorna VERDADERO únicamente si a no es nulo.
<b>( ... )</b>	Los paréntesis se usan para definir el orden de ejecución de las operaciones.	<b>b&gt;3 AND (a=1 OR a=2)</b>

### 3.2.3.ORDER

El parámetro \$order determina como se deben ordenar los resultados, utilizando los valores de las columnas especificadas, se utiliza de forma similar al ORDER BY de SQL. La clasificación se puede realizar en orden ascendente o descendente, el orden por defecto es ascendente.

Por ejemplo, mostrar todas las notarías con código notaria en orden descendente:

[https://www.datos.gov.co/resource/7hb3-hfs7.json?\\$order=notaria%20DESC](https://www.datos.gov.co/resource/7hb3-hfs7.json?$order=notaria%20DESC)

```

< -> C https://www.datos.gov.co/resource/7hb3-hfs7.json?$order=notaria%20DESC
[{"direcci_n": "CALLE 122 No 15 -21 LOCAL 201", "localizaci_n_soocrata": {"type": "Point", "coordinates": [-74.043142, 4.700771]}, "nombre_notario": "GLORIA CECILIA ESTRADA DE TURBAY", "notaria": "77", "telefono": "6203346 - 6203512"}, {"direcci_n": "Avenida Calle 53 No. 73A-73", "localizaci_n_soocrata": {"type": "Point", "coordinates": [-74.107933, 4.672585]}, "nombre_notario": "WILLI VALEK MORA", "notaria": "76", "telefono": "2638855 - 2638761"}, {"direcci_n": "Avenida Suba No. 106-52", "localizaci_n_soocrata": {"type": "Point", "coordinates": [-74.06744, 4.693518]}, "nombre_notario": "RAMÓN ALBERTO LOZADA DE LA CRUZ", "notaria": "75", "telefono": "2716446"}, {"direcci_n": "Carrera 80 I No 61-15 Sur", "localizaci_n_soocrata": {"type": "Point", "coordinates": [-74.184108, 4.610305]}, "nombre_notario": "NATALIA PERRY TURBAY", "notaria": "74", "telefono": "77754201-7776485"}, {"direcci_n": "AVDA EL DORADO No 69 C 03 L 103", "localizaci_n_soocrata": {"type": "Point", "coordinates": [-74.10818, 4.659956]}, "nombre_notario": "VICTORIA BERNAL TRUJILLO", "notaria": "73", "telefono": "2105146 -47 EXT120"}, {"direcci_n": "CALLE 71 No 10- 53", "localizaci_n_soocrata": {"type": "Point", "coordinates": [-74.058752, 4.655817]}, "nombre_notario": "PATRICIA TÉLLEZ LOMBANA", "notaria": "72", "telefono": "6062929-2484986"}, {"direcci_n": "Calle 61 A No. 16-10", "localizaci_n_soocrata": {"type": "Point", "coordinates": [-74.068644, 4.620924]}, "nombre_notario": "CARLA PATRICIA OSPINA RAMÍREZ", "notaria": "71", "telefono": "2483773 - 2350244"}, {"direcci_n": "CRA 69 J No 72 - 26", "localizaci_n_soocrata": {"type": "Point", "coordinates": [-74.090295, 4.682337]}, "nombre_notario": "CARLOS FELIPE CASTRILLON MUÑOZ", "notaria": "70", "telefono": "4913699- 4913714"}, {"direcci_n": "Avenida Calle 134 No. 7B-83 LC 1", "localizaci_n_soocrata": {"type": "Point", "coordinates": [-74.029457, 4.711811]}, "nombre_notario": "MARIA INÉS PANTOJA PONCE", "notaria": "69", "telefono": "6260669-6152969"}, {"direcci_n": "CALLE 37 SUR No 78 H 33", "localizaci_n_soocrata": {"type": "Point", "coordinates": [-74.152361, 4.623191]}, "nombre_notario": "JORGE HERNANDO RICO GRILLO", "notaria": "68", "telefono": "2650922-2650933"}, {"direcci_n": "Avenida Calle 72 No. 81A - 64", "localizaci_n_soocrata": {"type": "Point", "coordinates": [-74.103813, 4.693671]}, "nombre_notario": "EDUARDO LUIS PACHECO", "notaria": "67", "telefono": "4305294-95"}]

```

Podemos cambiar el orden de la consulta modificando DESC por ASC, o simplemente omitiéndolo ya que por defecto el ordenamiento es ASC.

## 3.2.4.GROUP

SoQL proporciona una cantidad limitada de funcionalidades para la agrupación a través del parámetro \$group. Como condición este parámetro debe ser utilizado en conjunto con \$select. Por ejemplo, determinar cada notario en cuantas notarias trabaja.

[https://www.datos.gov.co/resource/7hnu-p99i.json?\\$select=nombre\\_notario,COUNT\(notaria\)&\\$group=nombre\\_notario](https://www.datos.gov.co/resource/7hnu-p99i.json?$select=nombre_notario,COUNT(notaria)&$group=nombre_notario)

```
← → ↻ https://www.datos.gov.co/resource/7hnu-p99i.json?$select=nombre_notario,COUNT(notaria)&$group=nombre_notario

[ {
  "nombre_notario": "HERMANN PIESCHACON F.",
  "count_notaria": "1"
},
{
  "nombre_notario": "MARIA INÉS PANTOJA PONCE",
  "count_notaria": "1"
},
{
  "nombre_notario": "MANUEL J CAROPRESE MÉNDEZ",
  "count_notaria": "1"
},
{
  "nombre_notario": "LINA MARIA RODRÍGUEZ",
  "count_notaria": "1"
},
{
  "nombre_notario": "ANDRÉS HIBER ARÉVALO",
  "count_notaria": "1"
},
{
  "nombre_notario": "MARIA AMPARO QUINTERO ARTURO",
  "count_notaria": "1"
}
]
```

Otras expresiones de agrupación son:

FUNCIÓN	TIPO DE DATOS COMPATIBLES	DESCRIPCIÓN
<b>SUM</b>	Número	Suma todos los valores en una agrupación.
<b>COUNT</b>	Todas	Cuenta el número de valores. Si los valores están en NULL no se cuentan.
<b>AVG</b>	Número	Busca el valor promedio de los números en la columna.
<b>MIN</b>	Número	Busca el valor mínimo de los números en esta columna.
<b>MAX</b>	Número	Busca el valor máximo de los números en esta columna.

### 3.2.5.HAVING

El parámetro \$having permite filtrar los resultados de una agregación utilizando operadores booleanos, de forma similar a la cláusula HAVING en SQL. Por ejemplo, cuales son los notarios que trabajan en más de una notaría.

[https://www.datos.gov.co/resource/7hnu-p99i.json?\\$select=nombre\\_notario,COUNT\(notaria\)%20as%20cantidad&\\$group=nombre\\_notario&\\$having=cantidad%3E1](https://www.datos.gov.co/resource/7hnu-p99i.json?$select=nombre_notario,COUNT(notaria)%20as%20cantidad&$group=nombre_notario&$having=cantidad%3E1)

```
[ {
  "nombre_notario" : "OSCAR ANTONIO HERNÁNDEZ G",
  "cantidad" : "2"
}, {
  "nombre_notario" : "FABIO CASTIBLANCO CALIXTO",
  "cantidad" : "3"
} ]
```

Los mismos operadores del *WHERE numeral 3.2.2* del presente documento, aplican para el parámetro HAVING.

### 3.2.6.LIMIT

El parámetro \$limit controla el número total de filas devueltas, por defecto es 1.000 registros por solicitud. Se puede utilizar ya sea solo, o con \$offset con el fin de ver los resultados en páginas, este parámetro se explica más adelante.

Por ejemplo, ver las 3 primeras notarias, ordenadas de menos a mayor por su número.

[https://www.datos.gov.co/resource/7hnu-p99i.json?\\$order=notaria&\\$limit=3](https://www.datos.gov.co/resource/7hnu-p99i.json?$order=notaria&$limit=3)

```
← → ↻ https://www.datos.gov.co/resource/7hnu-p99i.json?order=notaria&limit=3
[ {
  "nombre_notario" : "HERMANN PIESCHACON F.",
  "direcci_n" : "CALLE 16 No 4 -62",
  "notaria" : "1",
  "telefono" : "3423049",
  "localizaci_n_soocrata" : {
    "latitude" : "4.600958",
    "needs_recoding" : false,
    "longitude" : "-74.07001700000002"
  }
}, {
  "nombre_notario" : "LEOVEDIS LEIAS MARTÍNEZ",
  "direcci_n" : "CRA 13 No 64 - 29",
  "notaria" : "2",
  "telefono" : "2358799",
  "localizaci_n_soocrata" : {
    "latitude" : "4.6510923",
    "needs_recoding" : false,
    "longitude" : "-74.0632481"
  }
}, {
  "nombre_notario" : "MANUEL J CAROPRESE MÉNDEZ",
  "direcci_n" : "Calle 12B No.8A-34 LC 9 - 10 y 11",
  "notaria" : "3",
  "telefono" : "4789177 - 4789179",
  "localizaci_n_soocrata" : {
    "latitude" : "4.600616",
    "needs_recoding" : false,
    "longitude" : "-74.075382"
  }
}
]
```

También es posible combinar la agregación \$group con \$limit, para ello se aplica el LIMIT después de la agregación.

### 3.2.7.OFFSET

El \$offset parámetro se utiliza con mayor frecuencia en conjunto con el parámetro \$limit. El \$offset divide la consulta en páginas, comenzando por el cero (0). Por ejemplo, para recuperar la "cuarta página" de la consulta anterior.

<https://www.datos.gov.co/resource/7hnu-p99i.json?order=notaria&limit=3&offset=4>

```
← → ↻ https://www.datos.gov.co/resource/7hnu-p99i.json?$order=notaria&$limit=3&$offset=4

[ {
  "nombre_notario" : "FABIO CASTIBLANCO CALIXTO",
  "direcci_n" : "CRA 15 A No 120 - 63",
  "notaria" : "5",
  "telefono" : "7450597 3132909944 3125228217",
  "localizaci_n_soocrata" : {
    "latitude" : "4.69937",
    "needs_recoding" : false,
    "longitude" : "-74.04326"
  }
}, {
  "nombre_notario" : "MARIA AMPARO QUINTERO ARTURO",
  "direcci_n" : "CRA 9 No 69 - 31",
  "notaria" : "6",
  "telefono" : "3170100 - 3145641",
  "localizaci_n_soocrata" : {
    "latitude" : "4.652759",
    "needs_recoding" : false,
    "longitude" : "-74.059074"
  }
}, {
  "nombre_notario" : "LIGIA JOSEFINA ERAZO CABRERA",
  "direcci_n" : "Calle 12 B No. 8-39 ",
  "notaria" : "7",
  "telefono" : "2826565 - 2826563",
  "localizaci_n_soocrata" : {
    "latitude" : "4.608145",
    "needs_recoding" : false,
    "longitude" : "-74.070811"
  }
}
]
```

### 3.2.8. BUSQUEDAS

El parámetro \$q se utiliza para acceder a un índice de texto especial que se busca dentro del conjunto de datos. Se puede interpretar más como un motor de búsqueda de una consulta SQL.

Por ejemplo, busca todos los notarios que el nombre sea María.

[https://www.datos.gov.co/resource/7hnu-p99i.json?\\$q=maria](https://www.datos.gov.co/resource/7hnu-p99i.json?$q=maria)

```
← → ↻ https://www.datos.gov.co/resource/7hnu-p99i.json?$q=maria

[ {
  "nombre_notario" : "MARIA AMPARO QUINTERO ARTURO",
  "direcci_n" : "CRA 9 No 69 - 31",
  "notaria" : "6",
  "telefono" : "3170100 - 3145641",
  "localizaci_n_soocrata" : {
    "latitude" : "4.652759",
    "needs_recoding" : false,
    "longitude" : "-74.059074"
  }
}, {
  "nombre_notario" : "MARIA ANGELA BEATRIZ SANIN POSADA",
  "direcci_n" : "AV. CALLE 82 No 11 -62 LOCAL 02",
  "notaria" : "35",
  "telefono" : "6221636- 6224562-6220291",
  "localizaci_n_soocrata" : {
    "latitude" : "4.665035",
    "needs_recoding" : false,
    "longitude" : "-74.053495"
  }
}
]
```

### 3.2.9. QUERY

El parámetro \$query le permite combinar múltiples cláusulas SoQL en una sola sentencia, para mayor comodidad. Al igual que en SQL, cláusulas deben tener un orden específico, como sigue:

- SELECT
- WHERE
- ORDER BY
- GROUP BY
- LIMIT
- OFFSET

Tenga en cuenta que, a diferencia de SQL, no está la cláusula FROM. Por ejemplo, puede combinar \$select y \$where los parámetros de la forma siguiente:

[https://www.datos.gov.co/resource/7hnu-p99i.json?\\$query=SELECT%20direcci\\_n,%20nombre\\_notario,%20notaria%20WHERE%20notaria%20%3E%2040](https://www.datos.gov.co/resource/7hnu-p99i.json?$query=SELECT%20direcci_n,%20nombre_notario,%20notaria%20WHERE%20notaria%20%3E%2040)

```
← → ↻ https://www.datos.gov.co/resource/7hnu-p99i.json?$query=SELECT%20direcci_n,%20nombre_notario,%20notaria%20WHERE%20notaria%20%3E%2040

[
  {
    "nombre_notario": "OSCAR ANTONIO HERNÁNDEZ G",
    "direcci_n": "Avenida Calle 134 No. 7B-83 LC 1",
    "notaria": "69"
  },
  {
    "nombre_notario": "ALIRIO VIRVIESCAS CALVETE",
    "direcci_n": "CRA 15 No 75 -09",
    "notaria": "41"
  },
  {
    "nombre_notario": "JUAN CARLOS VARGAS JARAMILLO",
    "direcci_n": "CALLE 85 No 14 -53",
    "notaria": "42"
  },
  {
    "nombre_notario": "JUAN ENRIQUE NIÑO GUARÍN",
    "direcci_n": "Avenida 19 No. 109-40",
    "notaria": "43"
  },
  {
    "nombre_notario": "LUZ MARY CÁRDENAS VELANDIA",
    "direcci_n": "Avenida 15 No. 96-07",
    "notaria": "44"
  },
  {
    "nombre_notario": "EDUARDO ISSAC CAICEDO ESCOBAR",
    "direcci_n": "CRA 15 No 91-06",
    "notaria": "45"
  },
  {
    "nombre_notario": "HELIA LUZ ALTAMAR LOZANO",
    "direcci_n": "CALLE 100 No 17- 25 PISO 2",
    "notaria": "46"
  },
  {
    "nombre_notario": "RICARDO CUBIDES TERREROS",
    "direcci_n": "CALLE 101 No 45 A 32",
    "notaria": "47"
  }
]
```

## 3.3. BÚSQUEDAS GEOGRÁFICAS

Usando el SODA API, podemos realizar búsquedas geográficas, como se muestra a continuación.

### 3.3.1. WITHIN\_BOX (...)

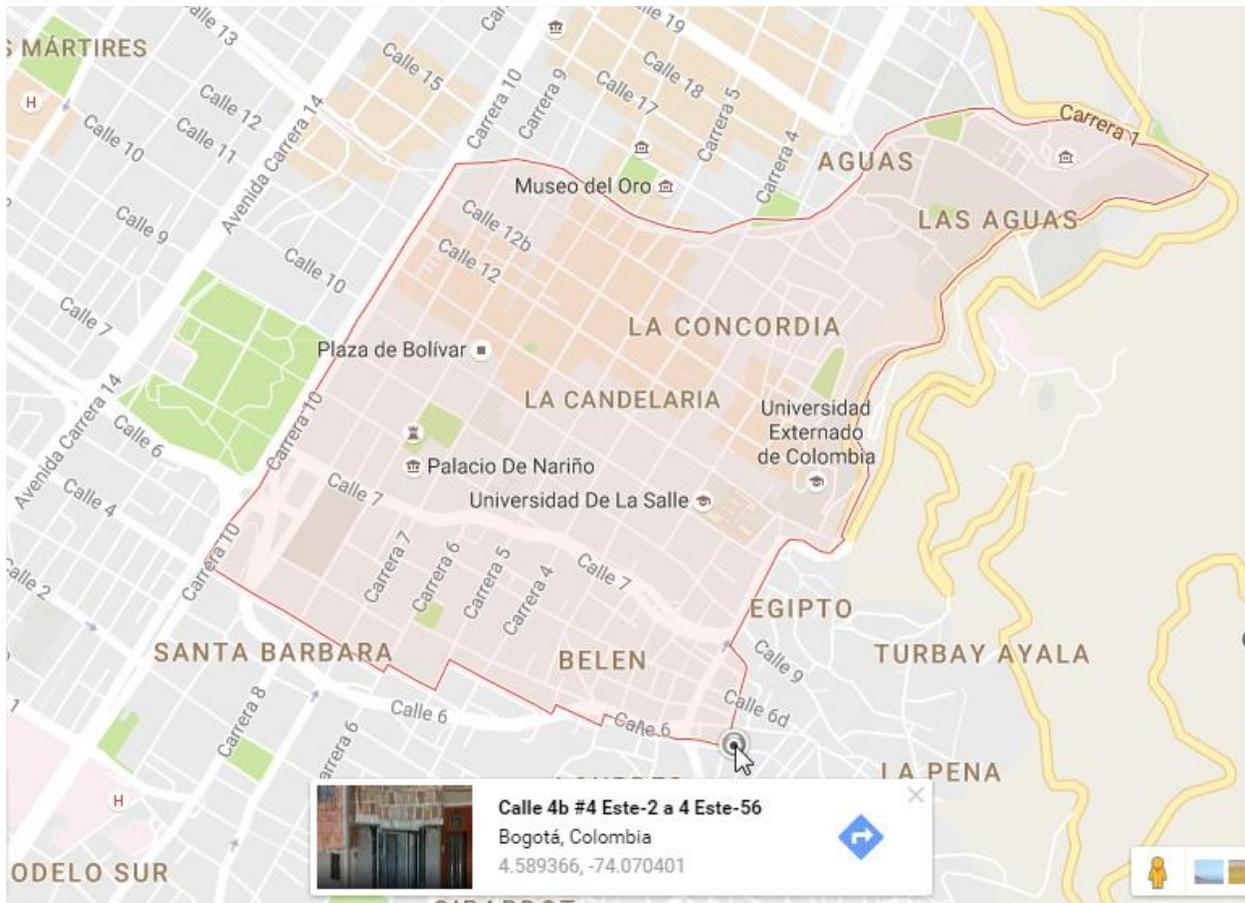
Devuelve las filas que tienen datos geográficos dentro de la “caja” o “box” definida por la latitud y la longitud. Esta búsqueda funciona con los siguientes tipos de datos:

- Ubicación.
- Punto.
- Multi-puntos (latitud, longitud)
- Línea.
- Polígono.

La función `within_box(...)` se usa conjunto con el parámetro `$where`, se filtran todos los puntos que se encuentren dentro de la “caja” definida por puntos. La función solicita 5 parámetros que se describen a continuación y se separan por coma (,) dentro de la función.

- El nombre de la columna del conjunto de datos que contiene el campo de búsqueda.
- La latitud de su punto noroeste.
- La longitud de su punto noroeste.
- La latitud de su punto sureste.
- La longitud de su punto sureste.

Por ejemplo, buscamos las notarías que se encuentren en el centro de Bogotá.



[https://www.datos.gov.co/resource/7hnu-p99i.json?\\$where=within\\_box\(localizaci\\_n\\_soocrata,4.614151,-74.075411,4.589366,-74.070401\)](https://www.datos.gov.co/resource/7hnu-p99i.json?$where=within_box(localizaci_n_soocrata,4.614151,-74.075411,4.589366,-74.070401))

```

[
  {
    "nombre_notario": "MANUEL J CAROPRESE MÉNDEZ",
    "direcci_n": "Calle 12B No.8A-34 LC 9 - 10 y 11",
    "notaria": "3",
    "telefono": "4789177 - 4789179",
    "localizaci_n_soocrata": {
      "latitude": "4.600616",
      "needs_recoding": false,
      "longitude": "-74.075382"
    }
  },
  {
    "nombre_notario": "LINA MARIA RODRÍGUEZ",
    "direcci_n": "Carrera 8 No.17-30 Piso 3, 4",
    "notaria": "4",
    "telefono": "6051313-2860629",
    "localizaci_n_soocrata": {
      "latitude": "4.604102",
      "needs_recoding": false,
      "longitude": "-74.073193"
    }
  },
  {
    "nombre_notario": "LIGIA JOSEFINA ERAZO CABRERA",
    "direcci_n": "Calle 12 B No. 8-39 ",
    "notaria": "7",
    "telefono": "2826565 - 2826563",
    "localizaci_n_soocrata": {
      "latitude": "4.608145",
      "needs_recoding": false
    }
  }
]

```

Que hace referencia a estos tres puntos que se muestran en el mapa del conjunto de datos.



### 3.3.2. WITHIN\_CIRCLE (...)

Devuelve las filas que tienen datos geográficos dentro de un círculo determinado en metros. Esta búsqueda funciona con los siguientes tipos de datos:

- Ubicación.
- Punto.
- Multi-puntos (latitud, longitud)
- Línea.
- Polígono.

La función `within_circle(...)` se usa conjunto con el parámetro `$where`, se filtran todos los puntos dentro de un radio determinado por un punto central. La función solicita 4 parámetros que se describen a continuación y se separan por coma (,) dentro de la función.

- El nombre de la columna del conjunto de datos que contiene el campo de búsqueda.
- La latitud de su punto central.
- La longitud de su punto central.
- El radio del círculo en metros.

Por ejemplo, buscamos las notarías que se encuentren a 500 metros del Parque Lourdes de la ciudad de Bogotá.



[https://www.datos.gov.co/resource/7hnu-p99i.json?\\$where=within\\_circle\(localizaci\\_n\\_soocrata,4.649351,-74.061707,500\)](https://www.datos.gov.co/resource/7hnu-p99i.json?$where=within_circle(localizaci_n_soocrata,4.649351,-74.061707,500))

```
← → ↻ https://www.datos.gov.co/resource/7hnu-p99i.json?$where=within_circle(localizaci_n_soocrata,4.649351,-74.061707,500)

[ {
  "nombre_notario": "MARIA AMPARO QUINTERO ARTURO",
  "direcci_n": "CRA 9 No 69 - 31",
  "notaria": "6",
  "telefono": "3170100 - 3145641",
  "localizaci_n_soocrata": {
    "latitude": "4.652759",
    "needs_recoding": false,
    "longitude": "-74.059074"
  }
}, {
  "nombre_notario": "GUILLERMO AUGUSTO ARCINIEGAS MARTINEZ",
  "direcci_n": "CALLE 63 No 10- 83 PISO 2",
  "notaria": "9",
  "telefono": "6401919 - 5420742 - 5430741",
  "localizaci_n_soocrata": {
    "latitude": "4.648428",
    "needs_recoding": false,
    "longitude": "-74.061048"
  }
}, {
  "nombre_notario": "JAIME ALBERTO RODRIGUEZ CUESTAS",
  "direcci_n": "Carrera 13 No. 63 - 39, IN 8 y 10",

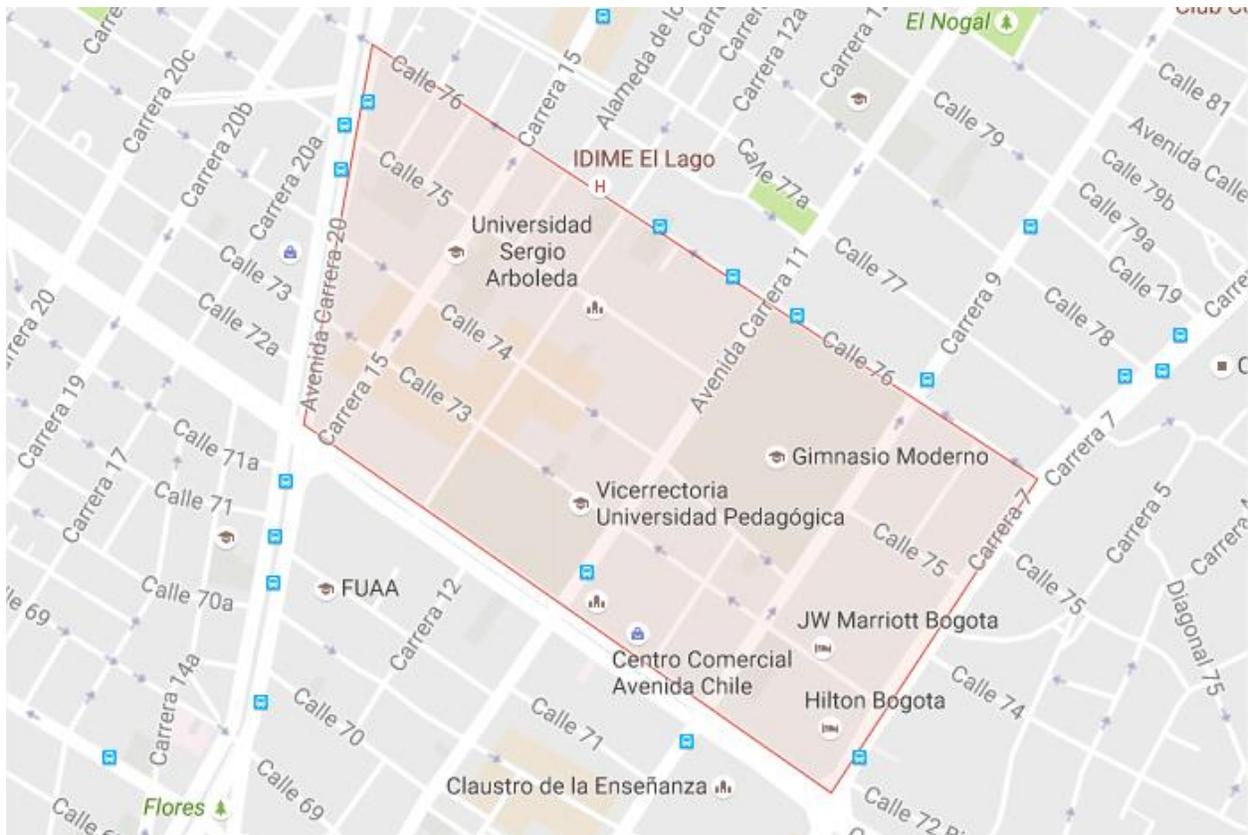
```



- El nombre de la columna del conjunto de datos que contiene el campo de búsqueda.
- Los puntos que definen el polígono, siguiendo el “Well-Know Text” un estándar de codificación de datos geospaciales de manera textual. Para más información de este estándar se puede consultar el siguiente link [https://en.wikipedia.org/wiki/Well-known\\_text](https://en.wikipedia.org/wiki/Well-known_text).

Ejemplo del estándar: MULTIPOLYGON ((-87.637714 41.887275, -87.613681 41.886892, -87.625526 41.871555, -87.637714 41.887275))

Por ejemplo, buscamos las notarías que se encuentren en el barrio La Porciúncula 500 de la ciudad de Bogotá.



[https://data.cityofchicago.org/resource/yama-9had.json?\\$where=within\\_polygon\(location,%20%27MULTIPOLYGON%20\(\(-87.637714%2041.887275,%20-87.613681%2041.886892,%20-87.625526%2041.871555,%20-87.637714%2041.887275\)\)\)%27\)](https://data.cityofchicago.org/resource/yama-9had.json?$where=within_polygon(location,%20%27MULTIPOLYGON%20((-87.637714%2041.887275,%20-87.613681%2041.886892,%20-87.625526%2041.871555,%20-87.637714%2041.887275)))%27)

# 04

# DATASYNC

DataSync es una herramienta de Exportación, Transformación y Carga (ETL) ofrecida por Socrata para acceder a los datos sin necesidad de usar el entorno del portal de Socrata. Se puede descargar la última versión de la siguiente página web: <https://github.com/socrata/datasync/releases>. El archivo que descargaremos será un .jar ejecutable.

Tenemos 3 tipos de operaciones posibles con DataSync: Standard Job, Port Job y Metadata Job. Todas ellas permiten ser guardadas en archivos JSON para permitirnos ejecutarlas nuevamente y, una vez guardadas, exceptuando el caso del port job, tenemos la posibilidad de generar un comando de tarea programada para configurarlo a través de nuestro sistema operativo y poder ejecutar la misma operación con la periodicidad que se desee.

Para ejecutar estas operaciones debemos introducir nuestros datos de autenticación de la plataforma de Socrata para que reconozca nuestros permisos sobre cada dominio. Debemos introducir el dominio con el que vamos a trabajar, nuestro correo electrónico de Socrata, la contraseña y nuestro Token.



Domain ?  
Username ?  
Password ?  
App Token ?



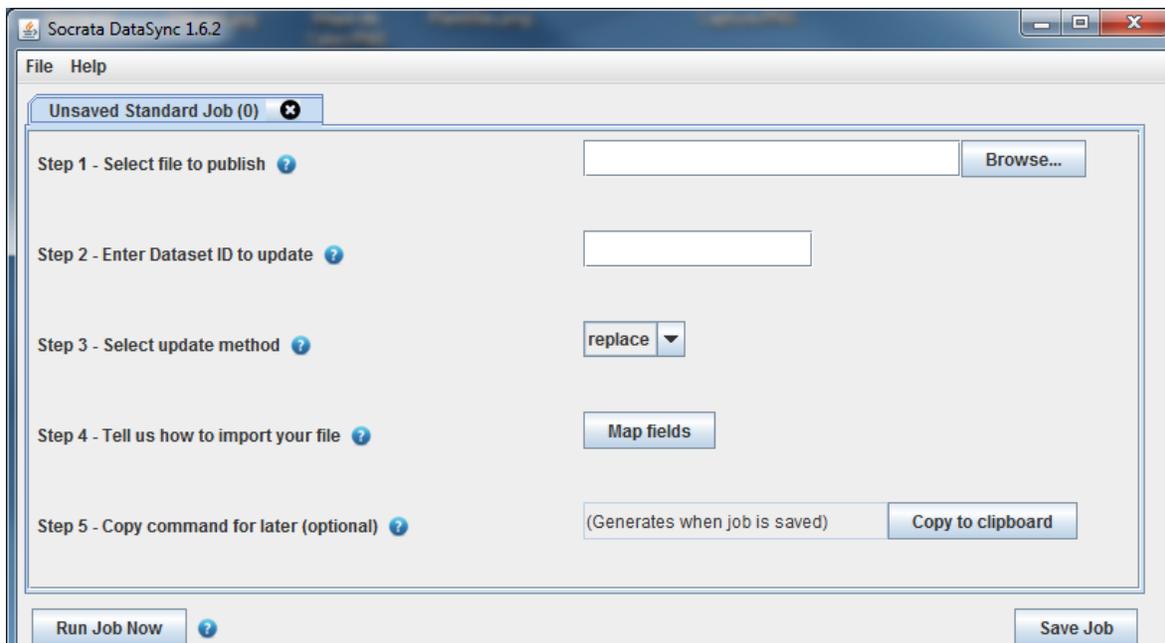
## 4.1. TOKEN PARA APLICACIONES

Socrata nos permite desde el perfil del usuario, generar un token con el fin de ser utilizado en la construcción de las aplicaciones, para que estas consulten los conjuntos de datos abiertos publicados en el portal del gobierno colombiano.

Para la generación exitosa de este token y poderlo usar, como veremos en los posteriores numerales, se puede consultar el siguiente link, el cual contiene un video tutorial paso a paso, que responde la pregunta de ¿Cómo genero un token para mis aplicaciones?, cualquier usuario puede generar token, no es necesario tener permisos específicos para este fin, el único prerequisite es contar con un usuario en el portal de los datos abiertos del gobierno colombiano, <https://www.youtube.com/watch?v=X5A2d5Yg7Qo>

## 4.2. STANDARD JOB

Esta operación permite acceder a un conjunto de datos en concreto para actualizar sus datos:



The screenshot shows the Socrata DataSync 1.6.2 application window. The title bar reads "Socrata DataSync 1.6.2". The menu bar includes "File" and "Help". Below the menu bar, there is a tab labeled "Unsaved Standard Job (0)". The main area contains five steps for configuring a job:

- Step 1 - Select file to publish: Includes a text input field and a "Browse..." button.
- Step 2 - Enter Dataset ID to update: Includes a text input field.
- Step 3 - Select update method: Includes a dropdown menu currently set to "replace".
- Step 4 - Tell us how to import your file: Includes a "Map fields" button.
- Step 5 - Copy command for later (optional): Includes a text input field with the text "(Generates when job is saved)" and a "Copy to clipboard" button.

At the bottom of the window, there are two buttons: "Run Job Now" and "Save Job".

Una vez seleccionamos, de nuestro equipo, el archivo .CSV o .TSV que queremos usar para actualizar el conjunto de datos e introducimos la id del conjunto de datos que queremos actualizar hemos de seleccionar una operación a elegir entre replace, append, delete y upsert.

- Replace: Reemplaza todo el conjunto de datos por el archivo.
- Append: Añade todos los registros del archivo al final del conjunto de datos, esta es una carga incremental de datos al conjunto de datos ya publicado en el portal de datos abiertos.
- Delete: Elimina el conjunto de datos.
- Upsert: Actualiza los registros que ya existen en el conjunto de datos y añade los que no existen.

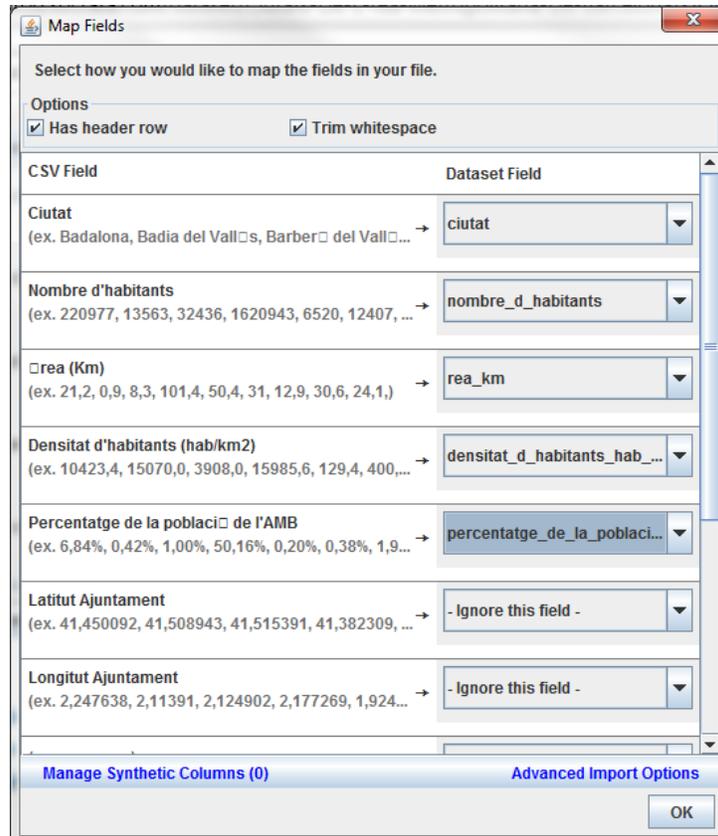
*Nota importante: Para usar la función upsert y que funcione correctamente hay que definir una columna que haga de "Punto Final de la API". Esto hará que considere esa columna como un identificador de fila y podrá recorrer el conjunto de datos tomando esa referencia. Debido a que se convertirá en un identificador de columna, no puede haber dos valores iguales en esa columna por lo que cada registro deberá tener ese valor distinto. En caso de no hacerlo, adjuntará todos los registros aún a riesgo de que existan registros repetidos. Para seleccionar la columna que hará de Punto Final de la API se hace en la ventana de definición de los metadatos:*

### Punto Final de API

Identificador de Fila

Ciudad

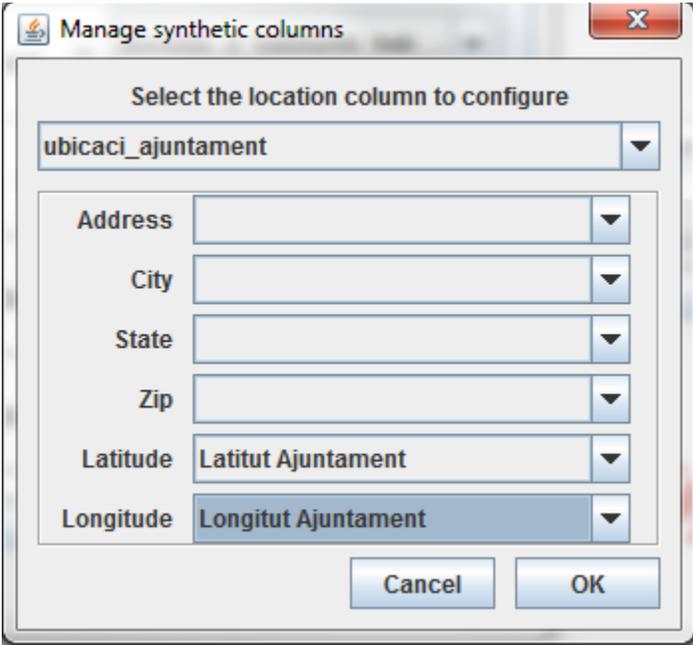
Una vez hayamos hecho esto, al clicar en Map fields podremos "mapear" las columnas del fichero con las columnas del conjunto de datos y acceder a las opciones:



En el ejemplo anterior hay dos temas a solucionar. Por una parte, no está mapeando correctamente las columnas de Latitud y Longitud ya que en el conjunto de datos del portal ambos datos aparecen de forma conjunta en un campo de localización y, por otra parte, existe un error de formato que se hace evidente en los acentos puesto que el formato de codificación por defecto es UTF-8 que no dispone de acentos y otros caracteres de Europa occidental. Para solucionar el primer problema hemos de hacer un mapeo de columnas sintéticas y para solucionar el segundo hemos de acceder a las opciones de importación avanzadas.

## 4.2.1. MAPEADO DE COLUMNAS SINTÉTICAS

Cuando en nuestro portal tenemos una columna sintética, es decir, que es fruto de varias columnas de la fuente original, con el DataSync debemos usar la opción “Manage Synthetic Columns”, y especificar la composición de dichas columnas, en el caso del ejemplo, vemos la columna de localización que es una mezcla de latitud y longitud. Una vez finalizada la configuración seleccionamos la opción “ok”.



## 4.2.2. OPCIONES DE IMPORTACIÓN AVANZADA

Advanced Import options

Timestamp Format

Separator

Quote

Encoding

Rows to skip

Escape

Timezone

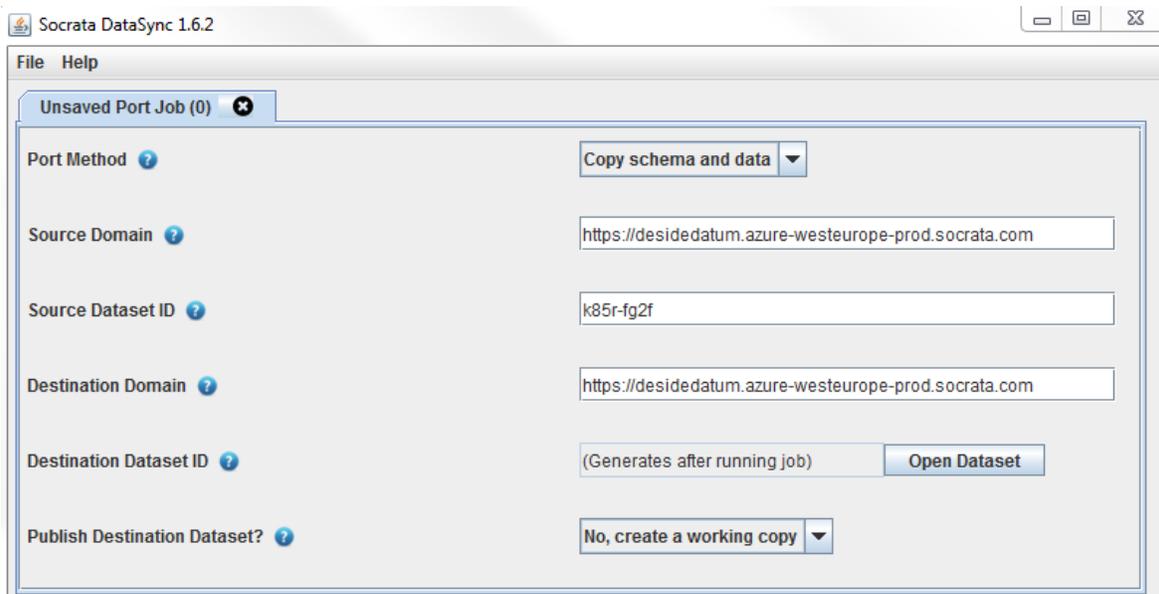
Empty text is null

Use Socrata Geocoding

Con las opciones de importación avanzada podemos configurar parámetros adicionales relativos a los formatos de datos y diferenciaciones regionales. Entre esas configuraciones, podemos cambiar el formato de codificación, por ejemplo, para resolver el problema que explicábamos antes. En este caso, cambiando UTF-8 por un formato que admita caracteres latinos como windows-1252 o ISO-8859-1 ya podríamos importar ese tipo de caracteres sin problema.

## 4.3. PORT JOB

El port-job nos permite copiar conjunto de datos. Podemos copiar los datos, el esquema o ambas cosas que supondría duplicar el conjunto de datos completamente:



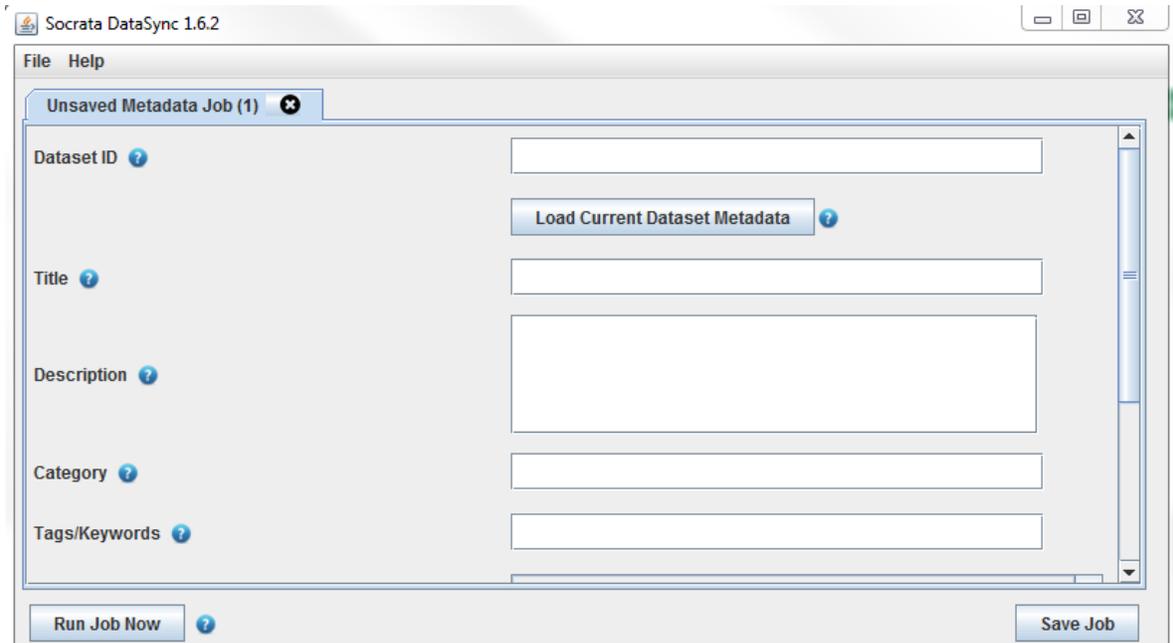
The screenshot shows the Socrata DataSync 1.6.2 application window. The title bar indicates the application name and version. The menu bar includes 'File' and 'Help'. A tab labeled 'Unsaved Port Job (0)' is active. The main area contains several configuration fields:

- Port Method**: A dropdown menu set to 'Copy schema and data'.
- Source Domain**: A text input field containing 'https://desidedatum.azure-westeurope-prod.socrata.com'.
- Source Dataset ID**: A text input field containing 'k85r-fg2f'.
- Destination Domain**: A text input field containing 'https://desidedatum.azure-westeurope-prod.socrata.com'.
- Destination Dataset ID**: A text input field containing '(Generates after running job)' and an 'Open Dataset' button.
- Publish Destination Dataset?**: A dropdown menu set to 'No, create a working copy'.

Para ejecutar esta operación, sólo necesitamos introducir el dominio fuente y el dominio destino, la id del conjunto de datos que queremos duplicar y esto nos generará la id del nuevo conjunto de datos creado una vez finalizada la operación.

## 4.4. METADATA JOB

El metadata job nos permite modificar los metadatos de un conjunto de datos. Simplemente hemos de introducir la id del conjunto de datos, cargar sus metadatos, actualizarlos a los valores que deseemos ejecutar la operación para actualizar los valores:



The screenshot shows the 'Socrata DataSync 1.6.2' application window. The main area is titled 'Unsaved Metadata Job (1)'. It contains several input fields for metadata: 'Dataset ID', 'Title', 'Description', 'Category', and 'Tags/Keywords'. A 'Load Current Dataset Metadata' button is positioned between the Dataset ID and Title fields. At the bottom of the form, there are two buttons: 'Run Job Now' and 'Save Job'. The interface includes a menu bar with 'File' and 'Help', and standard window controls (minimize, maximize, close) in the top right corner.

## 4.5. CONJUNTOS DE DATOS EN FORMATO XML

Si el conjunto de datos, se encuentra en un formato XML, como lo muestra la siguiente imagen, y se desea cargar a la plataforma de datos abiertos del gobierno colombiano, será imposible realizar esta acción directamente, ya que la plataforma no soporta conjuntos de datos en formato XML.

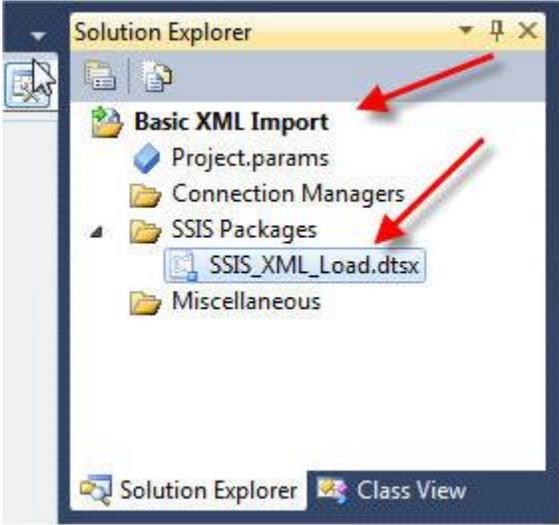
```
<?xml version="1.0" encoding="UTF-8"?>
<root>
  <row>
    <ANO>2000</ANO>
    <DIRECCION>ARAUCA</DIRECCION>
    <RENTA>359</RENTA>
    <IVA>632</IVA>
    <RETENCIONES>4,819</RETENCIONES>
    <EXTERNOS>5,494</EXTERNOS>
    <PORCLASIFICAR>109</PORCLASIFICAR>
    <SEGURIDADDEMOCRATICA>-</SEGURIDADDEMOCRATICA>
  </row>
  <row>
    <ANO>2001</ANO>
    <DIRECCION>ARMENIA</DIRECCION>
    <RENTA>7,944</RENTA>
    <IVA>17,189</IVA>
    <RETENCIONES>25,577</RETENCIONES>
    <EXTERNOS>-</EXTERNOS>
    <PORCLASIFICAR>44</PORCLASIFICAR>
    <SEGURIDADDEMOCRATICA>-</SEGURIDADDEMOCRATICA>
  </row>
  <row>
    <ANO>2002</ANO>
    <DIRECCION>BARRANCABERMEJA</DIRECCION>
    <RENTA>1,578</RENTA>
    <IVA>4,199</IVA>
    <RETENCIONES>8,234</RETENCIONES>
    <EXTERNOS>-</EXTERNOS>
    <PORCLASIFICAR>20</PORCLASIFICAR>
    <SEGURIDADDEMOCRATICA>-</SEGURIDADDEMOCRATICA>
  </row>
</root>
```

Por lo anterior el archivo deberá primero convertirse a un formato entendible por la plataforma, es decir, CSV, XLS o XLSX; para después si ser cargado como un conjunto de datos utilizando la opción del numeral 2.3.2 *Importar un archivo de datos*.

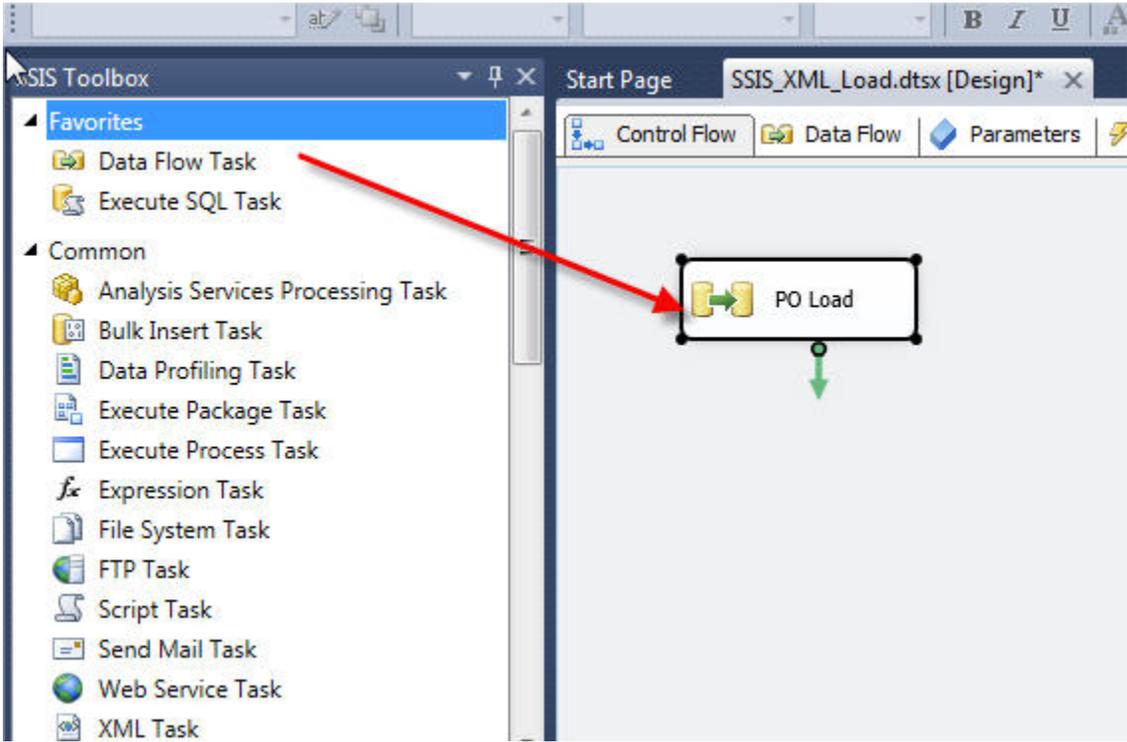
Para convertirlo podemos usar una ETL desde SSIS o la SODA API de Socrata, pero es fundamental dejar en claro que para que ambos procesos funcionen correctamente el archivo XML debe estar orientado a registros de filas y columnas, debemos omitir la utilización de caracteres espaciales, saltos de línea y delimitadores de campo.

## 4.5.1. UTILIZANDO SQL SERVER INTEGRATION SERVICES

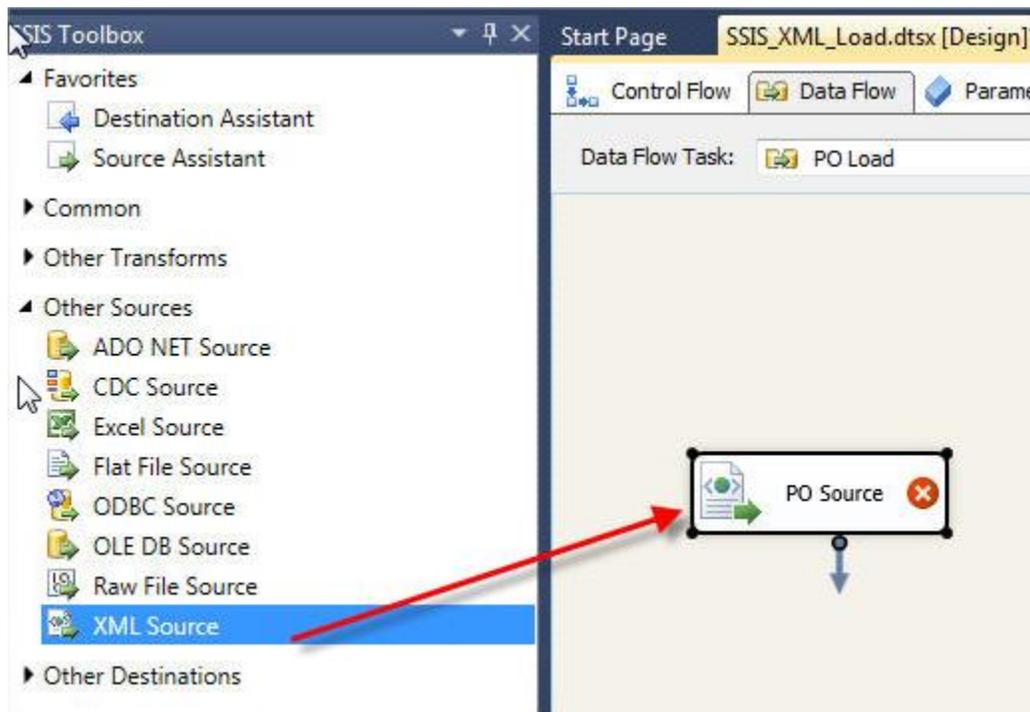
Se debe crear un nuevo proyecto SQL Server Data Tools, para este caso la hemos nombrado Basic XML Import.



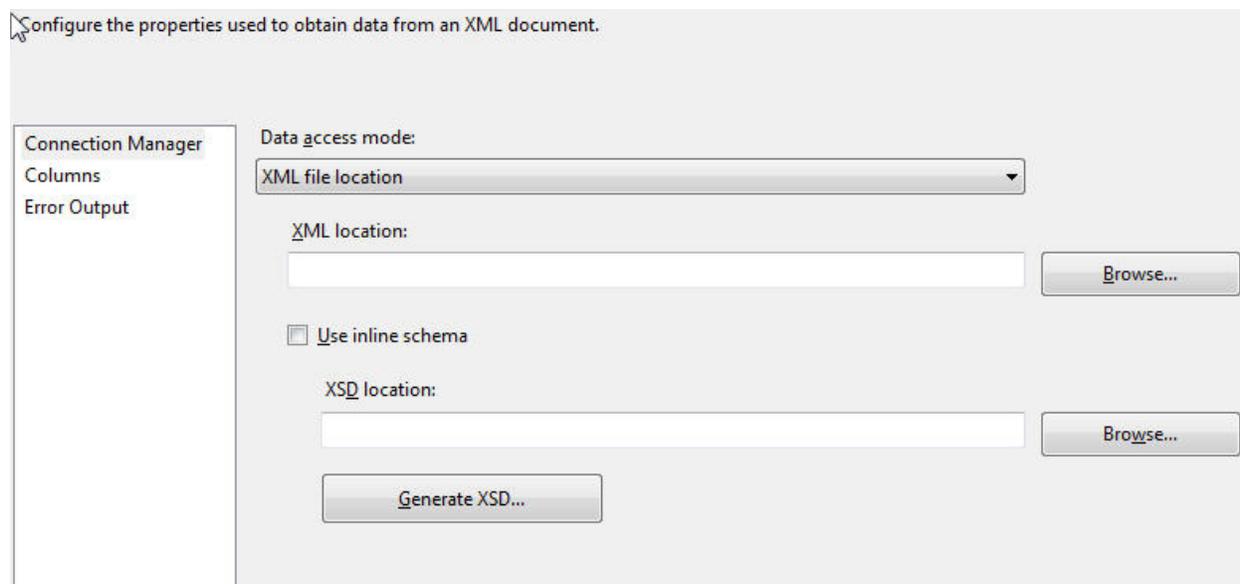
A continuación, insertamos un nuevo flujo o Flow Task, en el área de diseño, que para el caso del ejemplo se llama PO load.



Debemos definir las tareas que contendrá el flujo creado. Desde las herramientas debemos arrastrar al área de diseño “archivo XML”, la cual para este caso nombramos PO Surce.

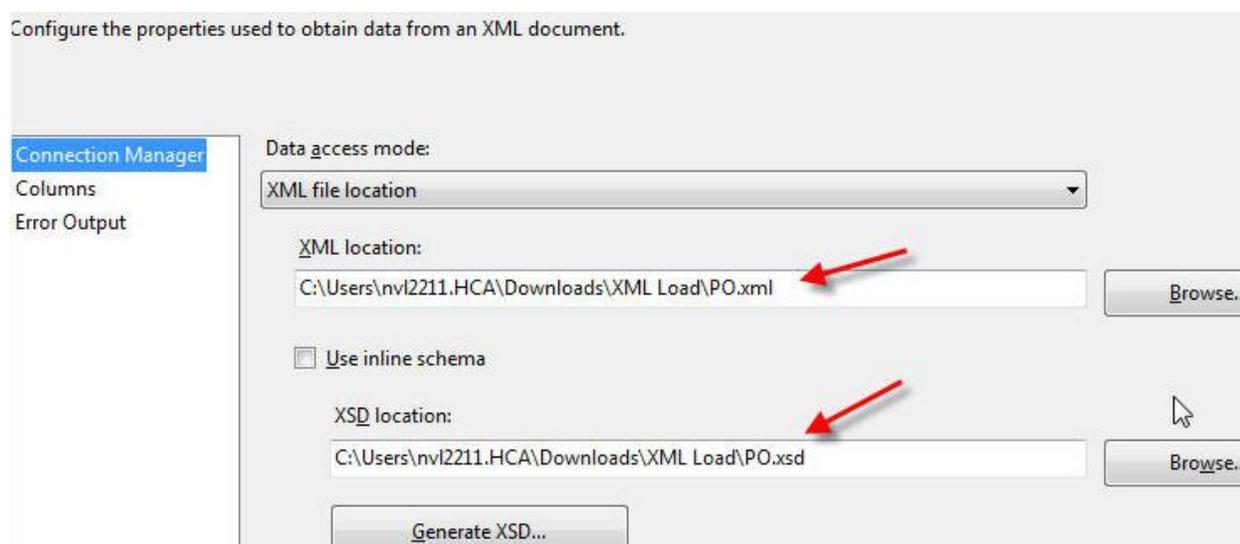


En las opciones de “archivo XML” en primer lugar debemos especificar la ubicación local del archivo como se muestra a continuación.

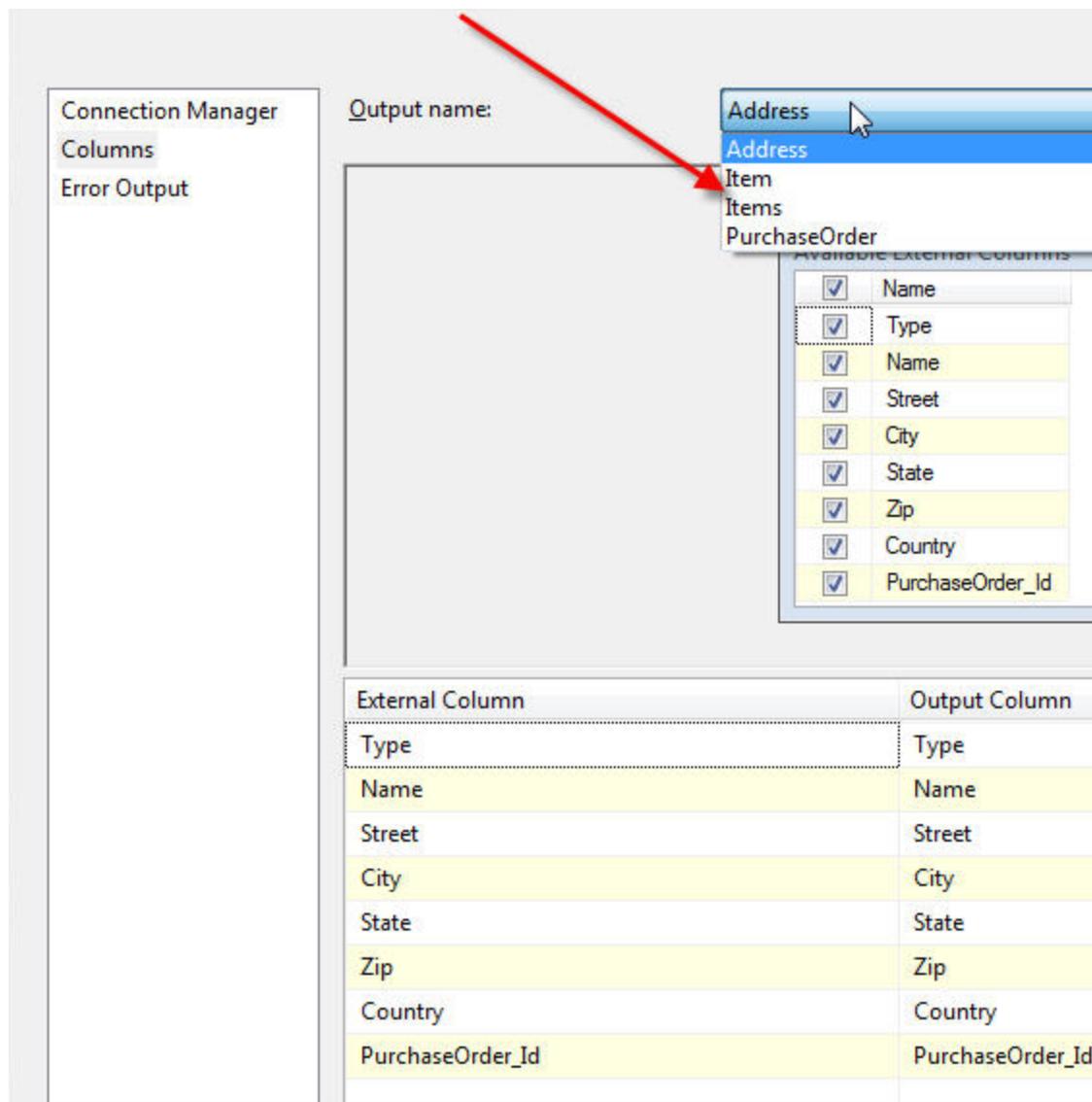


La opción “Use inline schema”, si no se selecciona esta opción por defecto estamos diciendo a SSIS que la definición del esquema XSD, se encuentra dentro del archivo XML, si no es así debemos seleccionar esta opción e indicar la ruta del XSD, este archivo expondrá en detalle las etiquetas dentro del archivo XML. En general, si se obtiene un archivo XML a partir de una fuente alternativa, el XSD ya debería estar integrado dentro del archivo XML.

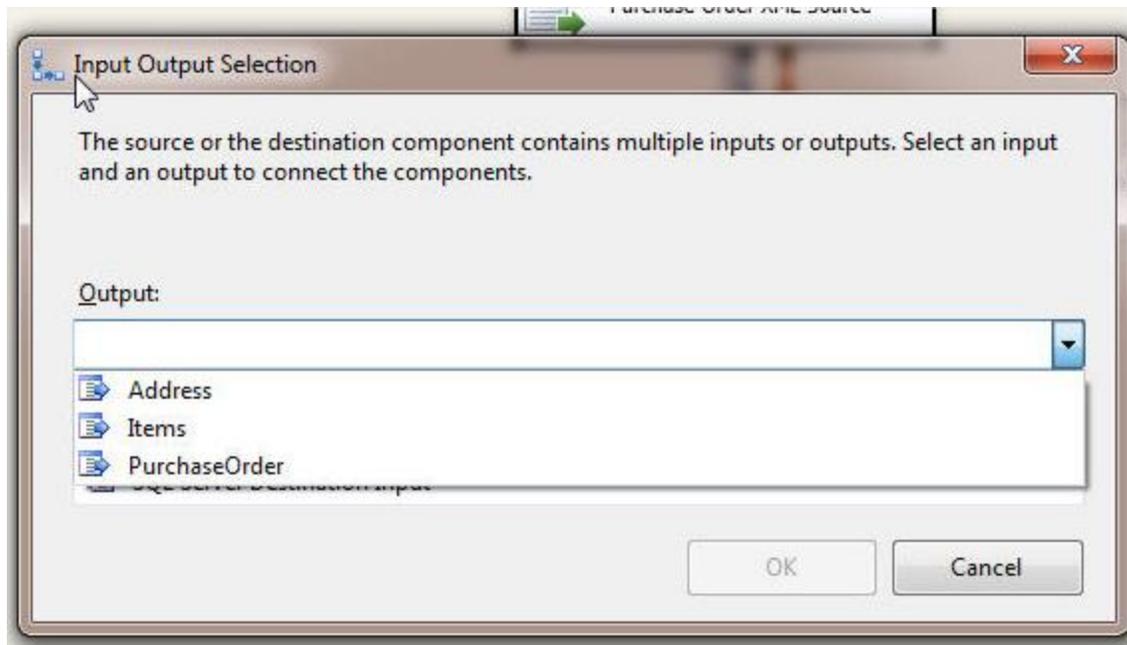
Para el ejemplo del tutorial, se está utilizando el archivo de órdenes de compra que se encuentra en el siguiente link <https://msdn.microsoft.com/en-us/library/bb387034.aspx> y para generar el XSD seleccionamos la opción “Generate XSD”.



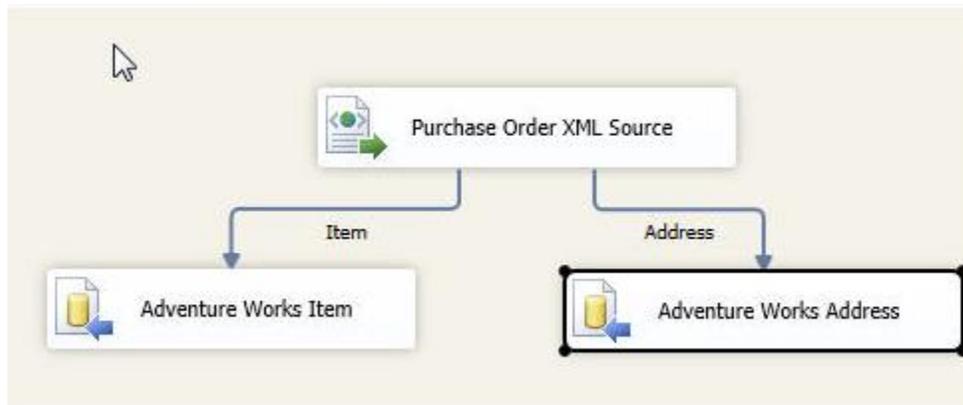
Una vez seleccionados los archivos, el siguiente paso es revisar las columnas potenciales a cargar. Para los archivos XML que contienen jerarquías y capas de elementos dentro de otros elementos, se genera un conjunto de tablas de datos independiente para cada elemento. En el ejemplo de archivo XML de órdenes de compra, también contiene las direcciones de envío “Address”, artículo (Item), artículos (Items) y elementos (PurchaseOrder).



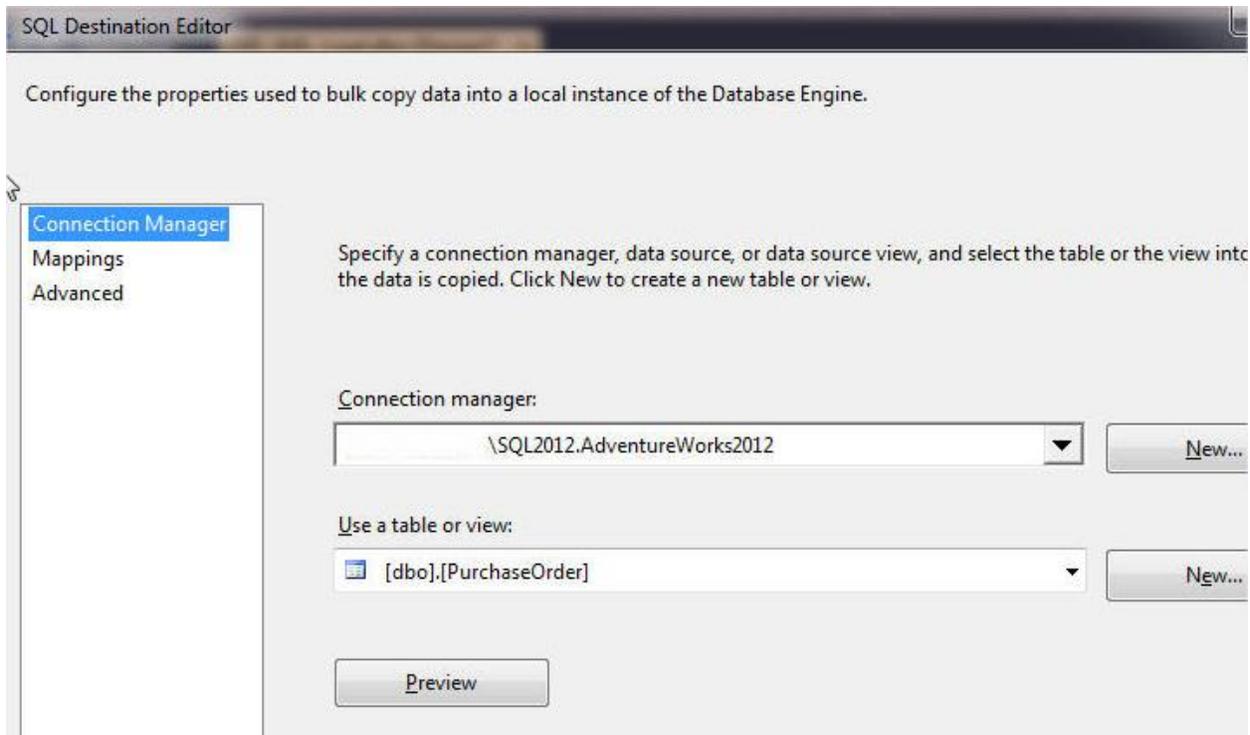
Para nuestro ejemplo PO, cada conjunto de tablas se va a cargar de forma individual. Por lo tanto, en realidad podemos añadir varios destinos de SQL Server, y dividir el flujo de datos, para ellos vamos a seleccionar la tabla XML a cargar, como se muestra a continuación.



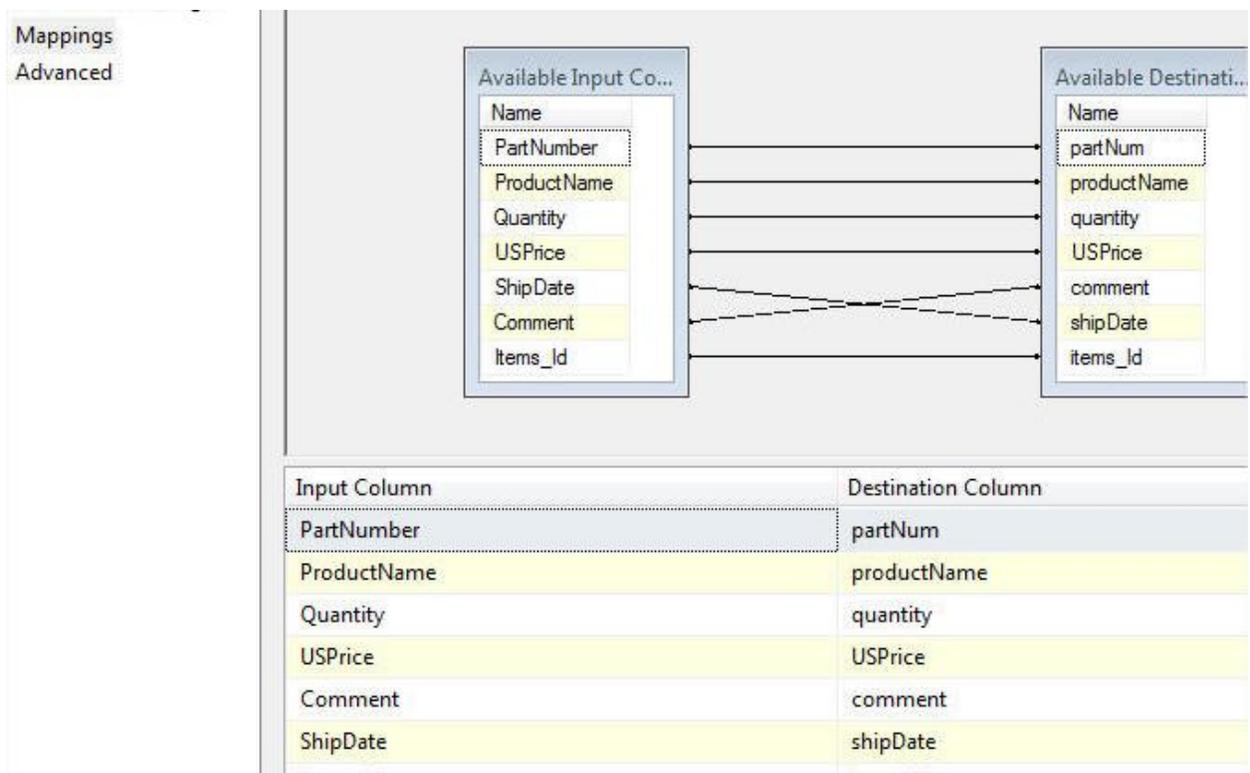
Siguiendo con nuestro ejemplo de órdenes de compra, la imagen a continuación muestra dos de flujos de datos que se cargaron en SQL Server. Observe cómo la trayectoria del flujo muestra la salida XML ubicada en la jerarquía más alta. Por supuesto, usted podría hacer lo mismo para la "línea roja" o flujo de error.



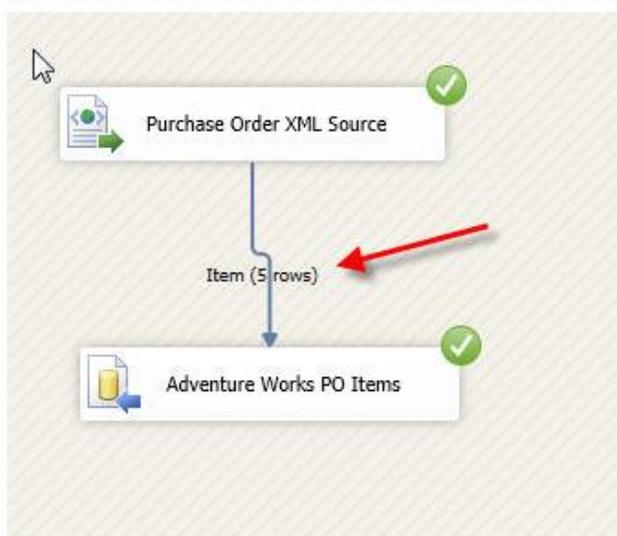
Nuestro último paso es mapear las columnas de origen a las columnas de la tabla destino de SQL Server. Se requiere la selección de la fuente de datos y nombre de la tabla donde se colocarán los datos.



A continuación, seleccionando la opción “Mappings” nos permite indicar la apropiada asignación de columnas como se muestra en la imagen a continuación.



De esta manera ya podemos ejecutar el proyecto y para el caso del ejemplo, el archivo XML solo contiene 5 artículos, los cuales se cargan con éxito en la tabla como se muestra a continuación.



	partNum	productName	quantity	USPrice	comment	shipDate	items_Id
1	872-AA	Lawnmower	1	148.9500000000	Confirm this is electric	NULL	4
2	926-AA	Baby Monitor	2	39.9800000000	NULL	1999-05-21 00:00:00.000	4
3	456-NM	Power Supply	1	45.9900000000	NULL	NULL	10
4	898-AZ	Computer Keyboard	1	29.9900000000	NULL	NULL	15
5	898-AM	Wireless Mouse	1	14.9900000000	NULL	NULL	15

Una vez la información este cargada en la tabla SQL podemos generar un siguiente flujo que convierta la información extraída en la tabla en un CSV delimitado por comas, para usar el DataSync como se ve en el numeral 3.2 en adelante, y cargar un conjunto de datos a partir de este CSV automáticamente, o ingresar a la plataforma de datos abiertos y cargar el CSV manualmente, para ello puede consultar el manual de publicador o el video tutorial de como publicar un conjunto de datos el cual se referencia en el siguiente link <https://youtu.be/B7JUVDfarKs>

## 4.5.2. UTILIZANDO SODA API

Utilizando el SODA API *numeral 2.3.1.1 de este manual*, propuesto por Socrata es posible desde un proyecto de JAVA leer el XML convertirlo en CSV y posteriormente utilizar el API para cargarlo a la plataforma. Utilizando una clase sencilla para manejo de archivos CSV.

```
public class CSVUtils {
    private static final char DEFAULT_SEPARATOR = ',';

    public static void writeLine(Writer w, List<String> values) throws IOException {
        writeLine(w, values, DEFAULT_SEPARATOR, ' ');
    }

    public static void writeLine(Writer w, List<String> values, char separators) throws IOException {
        writeLine(w, values, separators, ' ');
    }

    // https://tools.ietf.org/html/rfc4180
    private static String followCSVformat(String value) {
        String result = value;
        if (result.contains("\"")) {
            result = result.replace("\"", "\\\"");
        }
        return result;
    }

    public static void writeLine(Writer w, List<String> values, char separators, char customQuote) throws IOException {
        boolean first = true;
        // default customQuote is empty
        if (separators == ' ') {
            separators = DEFAULT_SEPARATOR;
        }
        StringBuilder sb = new StringBuilder();
        for (String value : values) {
            if (!first) {
                sb.append(separators);
            }
            if (customQuote == ' ') {
                sb.append(followCSVformat(value));
            } else {
                sb.append(customQuote).append(followCSVformat(value)).append(customQuote);
            }
            first = false;
        }
        sb.append("\n");
        w.append(sb.toString());
    }
}
```

Podemos seleccionar el XML, leerlo por cada objeto “row” y cada uno de estos elementos escribirlos en cada línea del CSV. A continuación, se muestra un ejemplo sencillo, de un loop for que recorre los objetos “row” y consulta cada uno de los tag, el resultado de la consulta se guarda en un objeto List, y con ayuda de la clase CSVUtils escribe las filas con base en el objeto List.

```

for (int temp = 0; temp < nList.getLength(); temp++) {
    Node nNode = nList.item(temp);

    System.out.println("\nCurrent Element :" + nNode.getNodeName());

    if (nNode.getNodeType() == Node.ELEMENT_NODE) {

        Element eElement = (Element) nNode;

        List<String> list = new ArrayList<>();
        list.add(eElement.getElementsByTagName("ANO").item(0).getTextContent());
        list.add(eElement.getElementsByTagName("DIRECCION").item(0).getTextContent());
        list.add(eElement.getElementsByTagName("RENTA").item(0).getTextContent());
        list.add(eElement.getElementsByTagName("IVA").item(0).getTextContent());
        list.add(eElement.getElementsByTagName("RETENCIONES").item(0).getTextContent());
        list.add(eElement.getElementsByTagName("EXTERNOS").item(0).getTextContent());
        list.add(eElement.getElementsByTagName("PORCLASIFICAR").item(0).getTextContent());
        list.add(eElement.getElementsByTagName("SEGURIDADDEMOCRATICA").item(0).getTextContent());

        CSVUtils.writeLine(writer, list);
    }
}
writer.flush();
writer.close();

```

```

1 ANO, DIRECCION, RENTA, IVA, RETENCIONES, EXTERNOS, PORCLASIFICAR, SEGURIDADDEMOCRATICA
2 2000, ARAUCA, 359, 632, 4819, 5494, 109, -
3 2001, ARMENIA, 7944, 17189, 25577, -, 44, -
4 2002, BARRANCABERMEJA, 1578, 4199, 8234, -, 20, -
5 2003, BARRANQUILLA, 71748, 144158, 263799, 423411, 339, -
6 2004, BUCARAMANGA, 35246, 77877, 115207, 3950, 102, -
7 2005, BUENAVENTURA, 11423, 6313, 10579, 985637, 7, -
8 2006, CALI, 162725, 436521, 605090, 244093, 412, -
9 2007, CARTAGENA, 24544, 55939, 100853, 1051840, 172, -
10 2008, CARTAGO, 3466, 3388, 5385, 2060, 23, -
11 2009, CUCUTA, 12316, 20239, 38483, 88672, 287, -
12 2010, ESPECIALGRANDES CONTRIBUYENTES, 1433929, 2117722, 3022949, -, 940, -
13 2011, ESPECIALPERSONAS JURIDICAS, 196542, 470134, 816339, -, 1623, -
14

```

Ya con el XML en formato CSV, podemos usar el API de Socrata para importar a base de un CSV un nuevo conjunto de datos con ayuda de la clase SodaImporter y el método createViewFromCSV.

```

// Instancia api producir el cual tiene insert, update, delete
SodaImporter importer = SodaImporter.newImporter(data.getUrl(), data.getUserSocrata(),
    data.getSocrataPass(), data.getToken());

// Create the dataset from the CSV
DatasetInfo createdView;

System.out.println("4. Cargando el nuevo DataSet en Socrata con el nombre de " + nombre);
createdView = importer.createViewFromCsv(nombre, descrip, folder);

```

Y posteriormente con ayuda de la clase DataSetInfo, podemos diligenciar la metadata del conjunto de datos.

```
Map<String, Map<String, String>> customMetadataToUpdate = new HashMap<String, Map<String, String>>();
customMetadataToUpdate.put(data.getTituloInfoEntidad(),mapaTituloInfoEntidad);
customMetadataToUpdate.put(data.getTituloInfoDatos(),mapaTituloInfoDatos);
Metadata metadata = new Metadata(customMetadataToUpdate, null, null, null, null);

// Actualizar la metadata
createView.setMetadata(metadata);
createView.setCategory(categoria);
createView.setTags(Arrays.asList(palabra_clave.split(" , ")));

importer.updateDatasetInfo(createView);
```

Al finalizar de diligenciar la metadata y tener en memoria en conjunto de datos, debemos publicarlo y darle permisos “públicos” para que pueda ser consultado por todos los usuarios.

```
idSocrata = createView.getId();
importer.publish(idSocrata);
importer.makePublic(idSocrata);
```

# 05

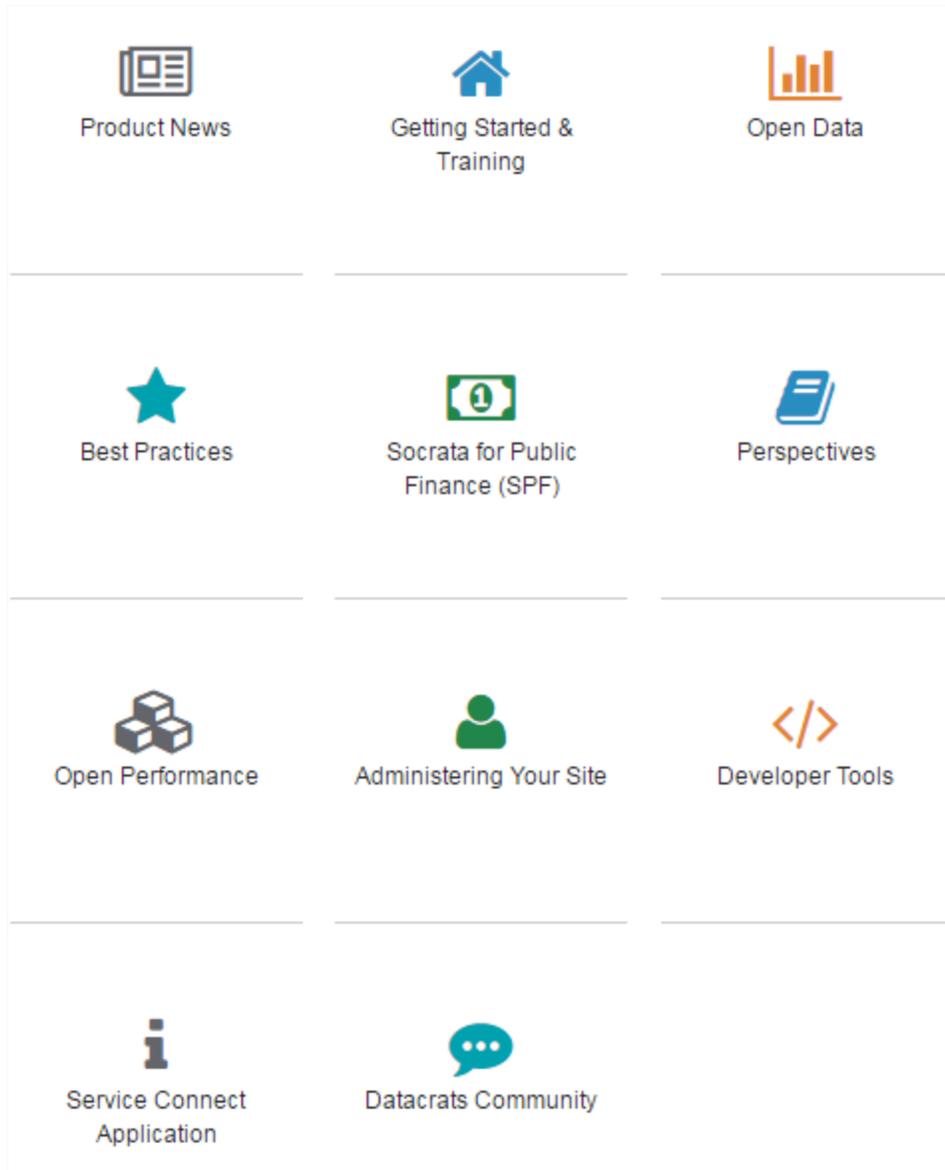
## SOPORTE DE SOCRATA

### 5.1 PÁGINA DE SOPORTE

Socrata ofrece una documentación que se encuentra en línea, disponible para cualquier ciudadano, <https://support.socrata.com/>, la página solo se encuentra en inglés.

The screenshot shows the top section of the Socrata support website. It features a dark blue header with a white lifebuoy icon on the left and the text "We Are Here To Help" in white. Below the text is a search bar with the placeholder "Search" and a magnifying glass icon. Underneath the header is a white navigation area. On the left, there is a promotional banner for an "Instructional Webinar: Make the Most of the Latest Features" with a subtext "Learn how to use new features for Open Data, Open Performance, Perspectives, and Socrata's Finance Suite." and an orange "Register & View Now!" button. Below the banner are four navigation items: "Product News" with a document icon, "Getting Started & Training" with a house icon, "Open Data" with a bar chart icon, and "All Systems Operational" with a green checkmark icon. To the right of "All Systems Operational" is a small promotional box for "Socrata Connect 2017" with the dates "March 6-8, 2017 | Washington, D.C." and the text "Save your seat now — and save 10%".

Donde podemos seleccionar entre una serie de categorías por tema, dependiendo de lo que queremos consultar.



La opción que más usaremos es “Open Data” donde encontraremos información sobre la creación y comportamiento de los lentes de datos, conjuntos de datos, filtros, graficas, calendarios o mapas, sección para todos los usuarios y como usar la plataforma y una sección de preguntas de ¿Cómo lo hago?

# OPEN DATA

## Data Lens

- ★ Data Lens Spotlight: Most Valuable Datasets in Data Lens
- ★ Creating Custom Choropleth Maps with Spatial Lens
- Finding a Dataset for Your Spatial Lens
- Using the Aggregation Function in Data Lens
- Roles & Permissions on Data Lens
- Saving & Publishing a Data Lens page

[See all 21 articles](#)

## For General Users

- Difficult CAPTCHA challenge
- Deleting a dataset or a row
- Reset your Socrata ID password
- Filter by category
- Fix log in problem resulting from Chrome AdBlock plug-in
- Suggest a dataset

[See all 9 articles](#)

## Datasets

- Hide a Column
- Deleting a row
- Add a new column
- Change dataset column width
- Change a column name
- Make dataset public or private

[See all 17 articles](#)

## Filters

- Find matches using 'is' and 'contains'
- Filter on AND/OR match conditions
- Dataset Grouping and Roll-Ups
- Use conditional formatting to enhance visualizations
- Sort a dataset or filtered view (Video Tutorial)

Dentro de las opciones Socrata nos ofrece explicaciones paso a paso, imágenes, noticias y videos, para complementar las explicaciones dadas en la página, recuerda que todo el material actualmente se encuentra únicamente en inglés.

# Exporting A Dataset For Excel

To download data in Excel format, you can export as a "CSV for Excel". CSV is a streamlined and expeditious format for data transfer and it will download quickly. After downloading, perform a 'Save As' locally to the format of your choice. Excel will open the CSV natively.



## 5.2 GENERAR UN TOKEN

Socrata provee de un servicio técnico a la disposición de todas las personas, para lo cual debemos estar registrados en la plataforma.

Ingresando a la página de soporte en la esquina superior derecha seleccionamos el botón "Sign in" y posteriormente "Sign up"

Sign in to Socrata Knowledge Base

 Sign in with Twitter

Email

Password

Stay signed in

**Sign in**

Your credentials will be sent over a secure connection

Cancel

[I am an Agent](#)

[Forgot my password](#)

New to Socrata Knowledge Base? [Sign up](#)

Have you emailed us? [Get a password](#)

If you've communicated with our support staff through email previously, you're already registered. You probably don't have a password yet, though.

Nos solicitara un correo y un nombre de usuario.

Sign up to Socrata Knowledge Base

Please fill out this form, and we'll send you a welcome email to verify your email address and log you in.

Your full name \*

Capacitación MINTIC

Your email \*

capacitacionmintic@gmail.com

Your Twitter

I'm not a robot

 reCAPTCHA  
Privacy - Terms

**Sign up**

cancel

Nos envía un correo al email que indicamos en el formulario, y debemos ingresar al link que allí envían para dar de alta al usuario. Allí nos solicitara la creación de la contraseña para la página de soporte de Socrata.

Elija una contraseña privada

Esta contraseña se usa para iniciar sesión en Socrata Knowledge Base.

Nombre

Contraseña

Requisitos de la contraseña:

- debe tener como mínimo 5 caracteres
- no puede ser igual a la dirección de correo electrónico

**Establecer contraseña**

Ahora cada vez que deseemos realizar una consulta al servicio técnico de Socrata, iniciamos sesión con el correo y contraseña indicados en los pasos anteriores, y en la esquina superior derecha de la pantalla seleccionamos “Submit a ticket”, donde le indicaremos si es una pregunta o un problema, o una consulta.

## SUBMIT A REQUEST

Please choose a request type below

-
Question or Issue
Feedback 

Dependiendo de la opción escogida nos muestra un formulario que debemos diligenciar en inglés, por medio de un correo nos responderán cuando Socrata haya contestado nuestra solicitud.