making business intelligence easy

Connecting JSON

Monday, April 10, 2017

JSON is (JavaScript Object Notation) is an open-standard format that uses human-readable text to transmit data objects consisting of attribute–value pairs. It is the most common data format used for asynchronous browser/server communication, largely replacing XML, and is used by AJAX.

JSON is a language-independent data format. It was derived from JavaScript, but as of 2017 many programming languages include code to generate and parse JSON-format data. The official Internet media type for JSON is application/json.
JSON filenames use the extension .json.

Yellowfin makes connecting to JSON data and sharing insights easy:

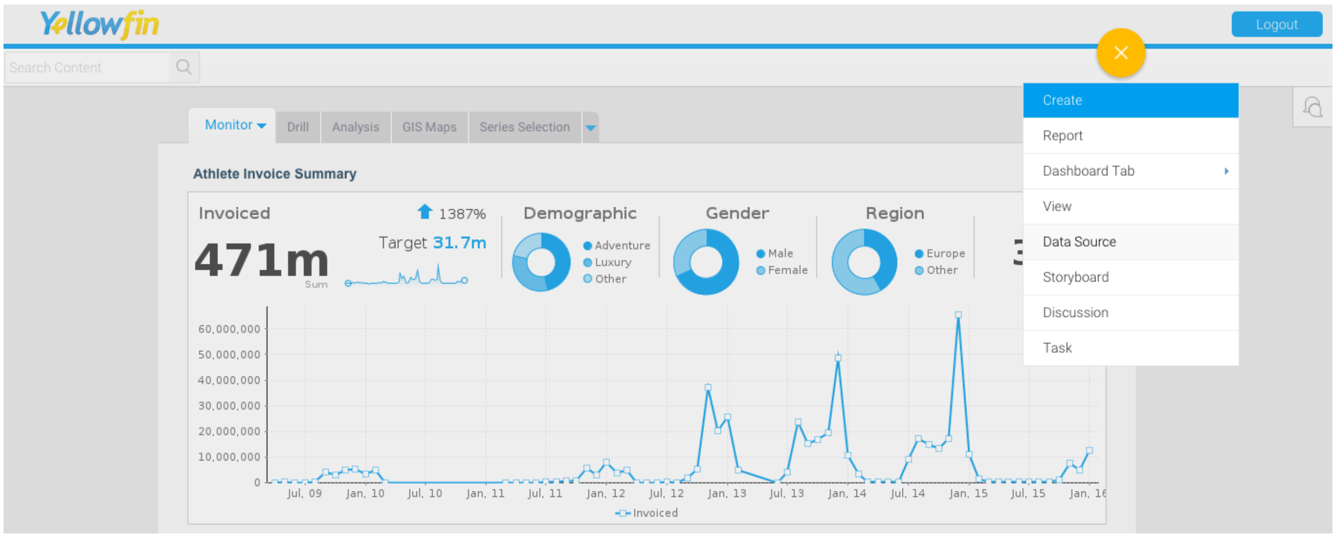
1. Install your connector
2. Select your JSON data source (File URL, FTP streaming or RESTful service)
3. Pick only the JSON keys you need

Install your Connector

Add a New Data Source

a) Log into your Yellowfin Server
(The User must have permissions to add a new Data Source)

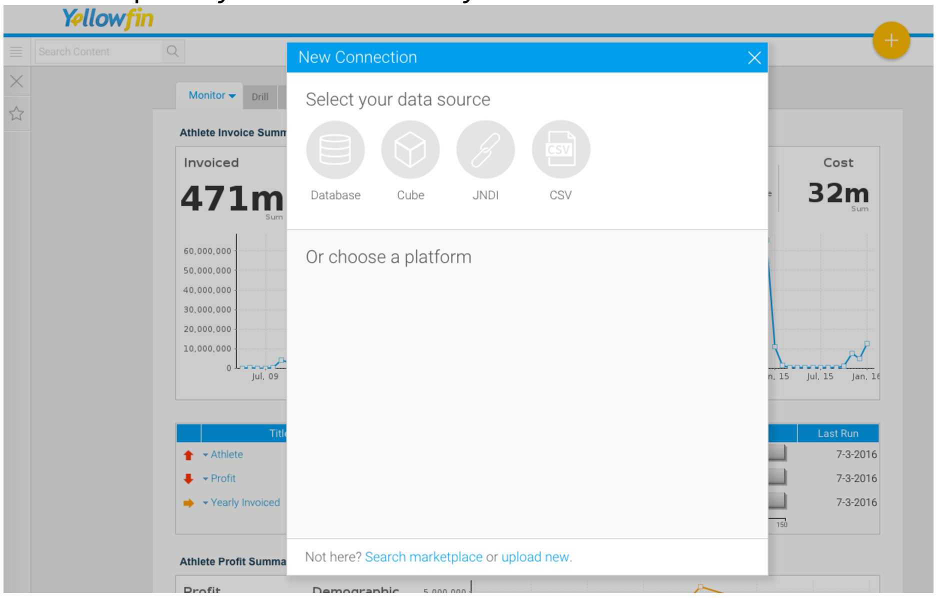
b) Click **Create** and select **Data Source**



Upload new connector

If you can see the icon for the JSON Connector at the **New Connection** step, then you can skip to Connect to your account. Otherwise you will need to select the **Not here?** links.

Search Marketplace - download your connector from the Yellowfin Marketplace
Upload New - upload your connector to your instance of Yellowfin



New Connection

Select your data source

Database Cube JNDI CSV

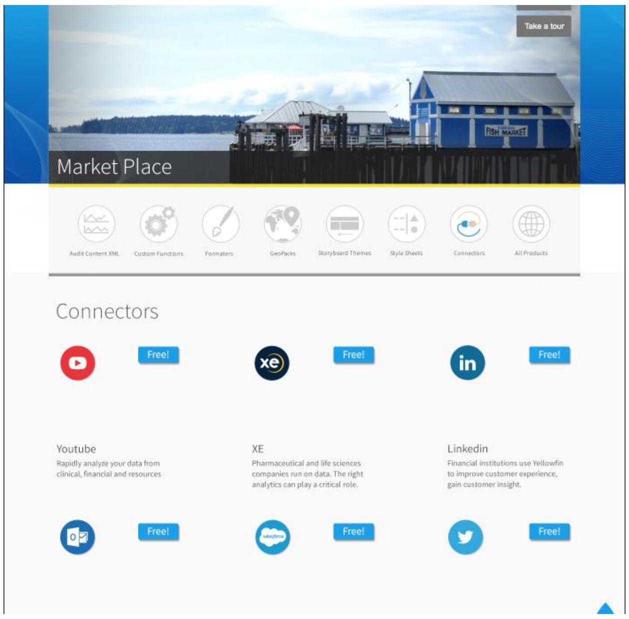
Or choose a platform

Not here? Search marketplace or upload new.

Download the connector from the Yellowfin Marketplace

c) Go to the [Yellowfin Marketplace]()

d) Download the **JSON Connector** (it's free!)



Connectors

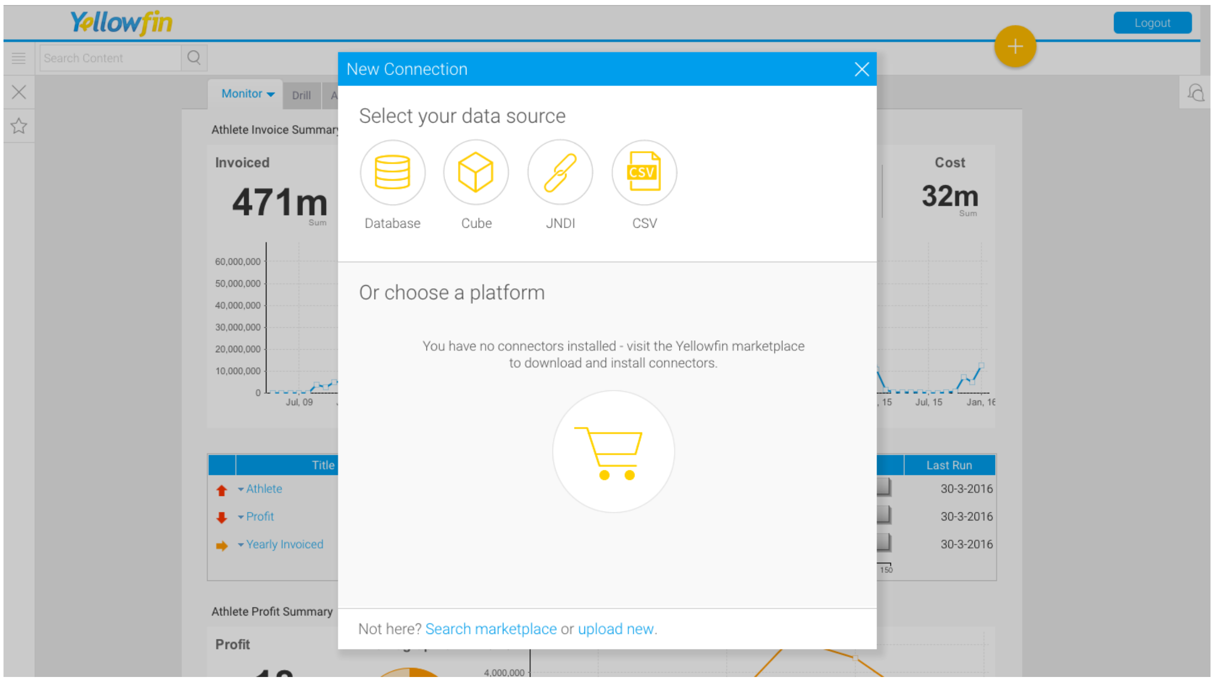
Youtube
Rapidly analyze your data from clinical, financial and resources

XE
Pharmaceutical and life sciences companies run on data. The right analytics can play a critical role.

Linkedin
Financial institutions use Yellowfin to improve customer experience, gain customer insight.

Upload your new Connector

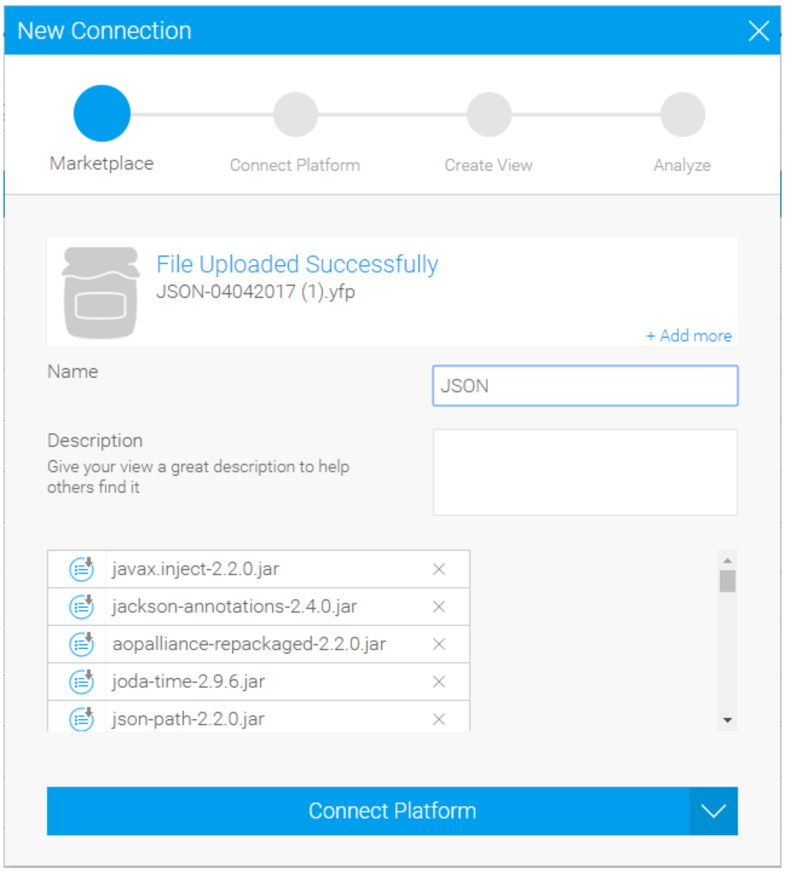
a) Click **Upload New**.


New Connection
Select your data source
Database
Cube
JNDI
CSV
Or choose a platform
You have no connectors installed - visit the Yellowfin marketplace to download and install connectors.
Not here? Search marketplace or upload new.

b) Give your connector a **name** and **description**.

c) Click **Connect Platform**.

Your connector will now be available under your **Data Source's** list.

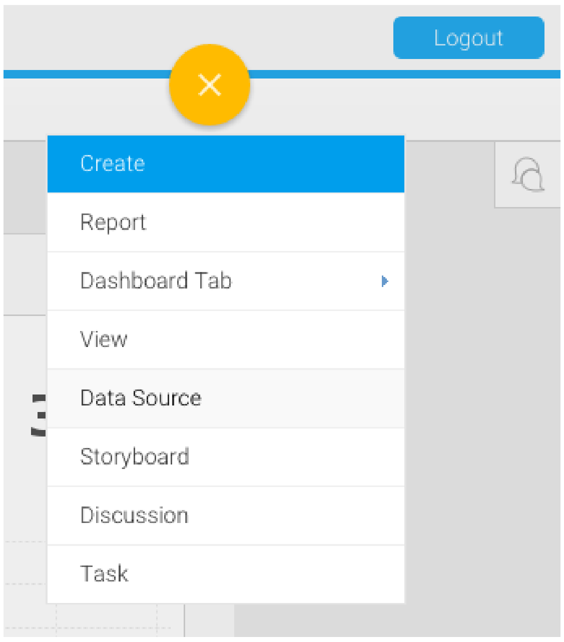


Connect to your account

Once your JSON Connector is installed, you can use it to

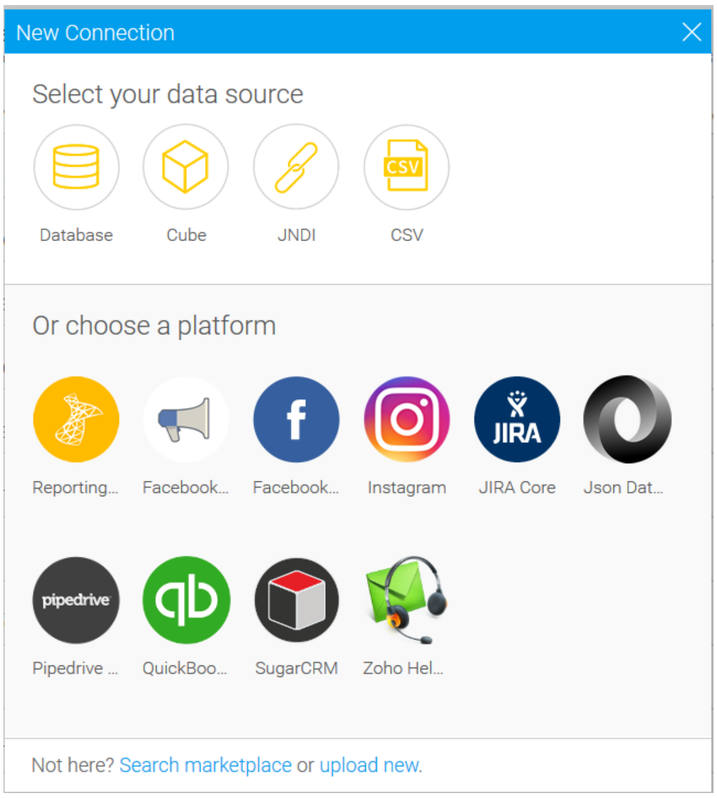
Create a new Data Source

a) Click **Create** and select **Data Source**.



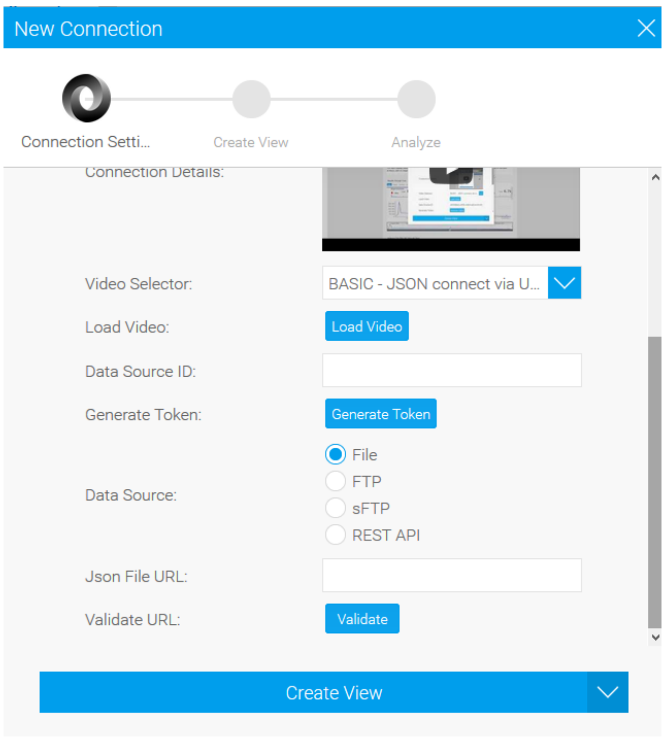
(If you have just installed your connector, then you will be immediately taken to the next step)

b) Select **JSON**



Select your JSON Source

URL
FTP
REST


New Connection

Connection Setti... | Create View | Analyze

Connection Details:

Video Selector: BASIC - JSON connect via U...

Load Video: Load Video

Data Source ID:

Generate Token: Generate Token

Data Source: [✓ File] FTP sFTP REST API

Json File URL:

Validate URL: Validate

Create View

Watch and Learn

Watch our "how to" videos to accelerate your skills with the JSON connector

1. **Consuming JSON data from a URL source**

   https://youtu.be/lQVTQ-RYJ3c

2. **Consuming JSON data from a FTP folder source (Streaming or Static)**

   https://youtu.be/ullq62taTeM

3. **Consuming JSON data from a static REST Source**

   https://youtu.be/67Hp-W-PiJU

4. **Working with JSON data that has separate Lat/Long keys**

   https://youtu.be/LR2gmKppP1s

5. **Setting up your JSON data source – JSON Formatters**

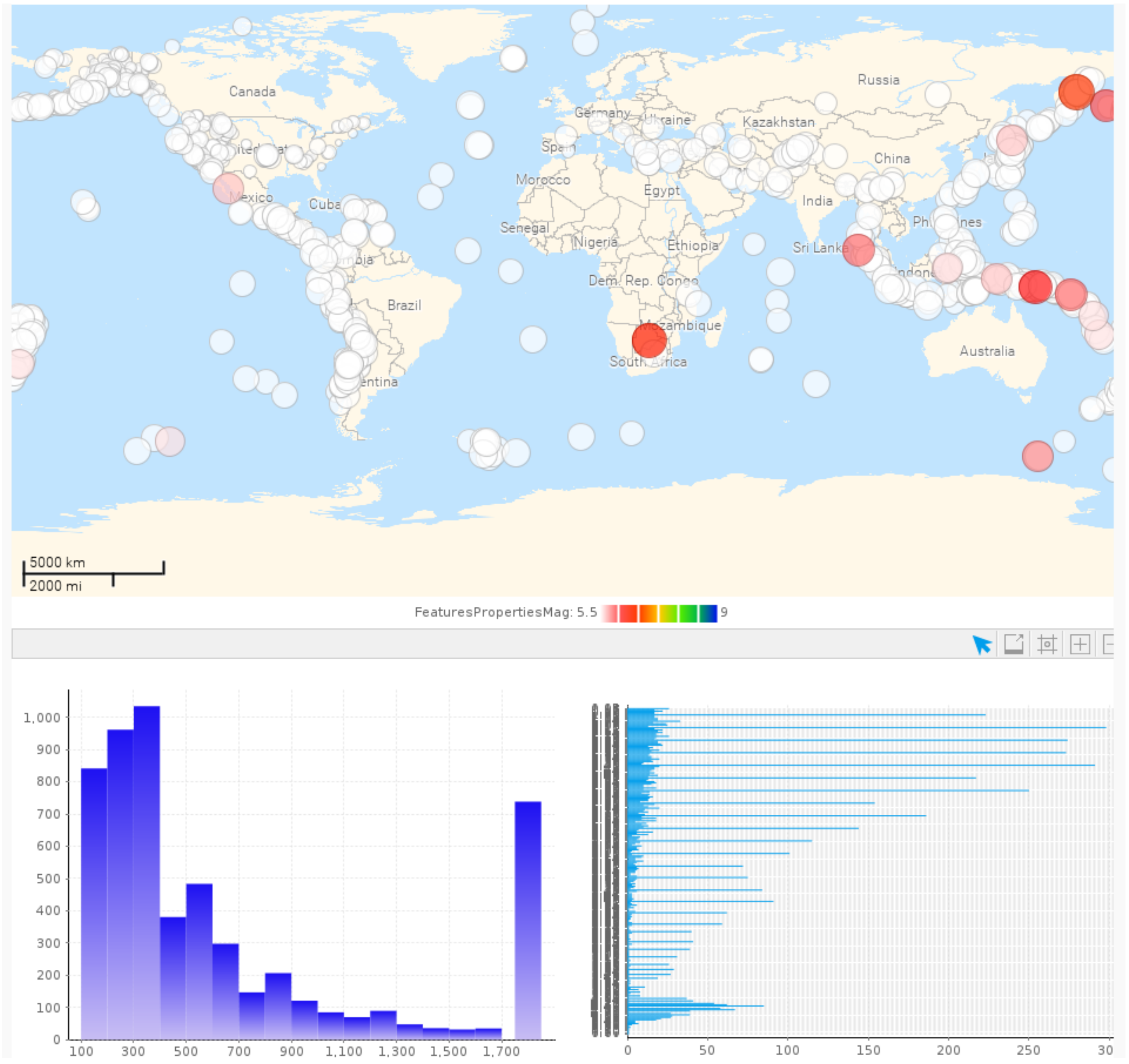
   https://youtu.be/PFegJQTzJns

6. **Consuming SODA with the Yellowfin JSON connector**

https://youtu.be/nhsor7wop4E

SODA Search Engine https://dev.socrata.com/data/

Explore the JSON World



Here are some JSON dataset links to get you started

1. (URL) US Geological Survey - Earthquakes

https://earthquake.usgs.gov/earthquakes/feed/v1.0/geojson.php

2. (REST) New York Prison Inmate intake (March 2017)

https://data.cityofnewyork.us/resource/gqrb-77i6.json

$where=admitted_dt%20between%20%272017-03-01T00:00:00%27%20and%20%272017-03-31T23:59:59%27

3. (REST) Location of Blueberry Farms in the US state of Connecticut

https://data.ct.gov/resource/y6p2-px98.json

$$app_token=blYkKCKxsaBGdtvwlpM4bQ4Iz;category=Fruit;item=Blueberries

4. (URL) NASA Meteorite landings

https://data.nasa.gov/view/ak9y-cwf9

JSON Data Sources

URL

The connector can access JSON formatted data from any available URL. A good example is the US geological survey earthquake data

https://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/4.5_month.geojson

The data may be static, or, in the case of the earthquake data, the url remains the same but the content is updated every 15 minutes. This allows the connector to provide constant fresh data views.

For example; If a user maps the location of earthquakes above 4.5 magnitude, in the last 7 days; each day they log into their dashboard, they will notice new incidence without having to alter the parameters of the report.

FTP/sFTP

FTP is standard plumbing everywhere, it is robust and well defined. But it is slow.
For each file read, the FTP protocol opens and closes a TCP/IP connection.
It is faster to process a single large file, than thousands of small files. However, for near real time or after hours processing it is reliable.

The connector can connect to a straight FTP source or alternatively to a sFTP (more secure FTP) source.

The speed of your FTP server and the network lag, will determine processing performance.

The connector supports file streaming. This is, it can continually look at the folder and process new files arriving on a DELETION or TIMESTAMP strategy.

In DELETE mode, the connector will read a file, process the JSON contents, Store the contents and then delete the original file from the source folder.
DELETE mode is clearest way to judge processing, as the only files left in the source folder will be files yet to be processed by the connector. This mode is best for internal company data, where the source files are a mirror copy of a source repository.

In TIMESTAMP mode, the connector keeps track of the files it has processed from the source folder. It will process the files in FIFO order and will not alter or delete the source files. TIMESTAMP mode is recommended, when the source files must not be altered by the connector or any other 3rd party application.

We have provided a LIMITER function, so that users can set the quanta of work for each scheduled run of the report. Hence, if you have 1000 files to be processed, you may set the limiter to 100 and increase the frequency in which the report is run. In this way, each time the report runs it will process only 100 waiting files, resulting in updates to the report.   This strategy works well for file streaming or having a large static number of files to process; while still being able to get some report insights.

REST

The connector can process data from a REST service.

The connector in the current version does not support data storage and pagination. This will be delivered in a future version.
The difficulty with pagination is that there is no universal approach to pagination; often with each service taking a different approach.

In this version, the connector can continually connect and process JSON formatted data from a REST service; but it is limited to the full data set it can access in a single call.

For very large JSON record sets, this will have a performance impact; We therefore, recommend that the user, utilizes filters in the parameters to select only the data topics of interest.

For example. With SODA, it is possible to select a dataset of all crime in New York from 2001 to 2017. This would be a very large dataset.
If the user if only interested in "Car theft" for the years 2010 to 2017. We recommend that the user applies date and crime type filters on the REST call to reduce the dataset returned and ensure good performance.

All SODA sources offer a comprehensive set of filters and other REST services generally have limit clauses to reduce the amount of data selected.

JSON Data Types

The JSON designer available in the connector, is a service to allow users to perform two main functions

1. PICK only the data elements that need
2. Set the correct type on the data

The designer will parse the complete (presented) JSON record set and provide the list of available keys. The user may select manually the keys to include from the list. Alternatively, the user may choose to select all keys and then delete the keys not required.

The designer offers key options and can manage nested JSON record sets.

When a user selects all keys automatically, the designer will prepopulate the column names with a CamelCase key path. This may be altered by the user overtyping the column name.

The designer can manage numeric values in native JSON numeric forms or numeric values that are wrapped as text. Users may alter a numeric value between integer or Numeric TYPE.

The designer can manage several date formats common in business and scientific JSON records. The designer, allows the user to propose a value is a DATE and then provided the exact format from a list of common date/time formats. If a common format is not available, the user may build the format with standard regex values.

The designer can manage several co-ordinate formats common in business and scientific JSON records.

The key abilities of the designer and connector is the ability to identify and format POINT formats (GEOJSON), or to combine separate latitude/longitude values into a POINT. For POINT values the user must convert to a WKT GIS format inside Yellowfin to realize the POINT for mapping charts or Google maps.

FAQ.

[1] In sFTP mode the source folder is reporting an error?

Consider you want to collect files from a subfolder on your FTP server.

Called → TAXFiles

In FTP mode, you just need to enter   TAXFiles   in the FTP/sFTP Folder field
In sFTP mode, you just need to enter   /TAXFiles   in the FTP/sFTP Folder field

The difference is the forward slash in sFTP mode.

[2] In REST mode or URL mode, do you support LineStrings used to plot paths or tracks ?
e.g.

```
Railroad","rrowner_1":"XXXX","shape_len":"1122.66789119","status":"ABANDONED","s
ubdivisio":"ALAMOSA","the_geom":{"type":"LineString","coordinates":[[-
104.85245975757725,37.679459753654896],[-
104.85349875396625,37.68094875282956],[-104.85471075736902,37.68268675209258],[-
104.85479776219142,37.68289275187236],[-104.85514575664617,37.68474675394367],[-
104.85517575547097,37.685364747848894],[-
104.8553777603809,37.685913748050055],[-104.85578175664907,37.68655475118986],[-
104.85676175398726,37.68736775304521],[-104.8570507627139,37.687606750783765],[-
104.85765675651535,37.68783474694732],[-104.85820376273219,37.68790275383686]]}}
```

No, the connector will only pick up the first POINT in the collection.
Processing paths is not the usual best use of a BI tool, we recommend you look at a purpose GIS mapping tool to work with LineStrings.

[3] How do I search the SODA network of public datasets for JSON sources ?

Datasets may be searched via the following search link

https://dev.socrata.com/data/

We recommend you watch video 6 on how to search and consume the JSON datasets using the Yellowfin JSON connector.

[4] How do I search a date range using the SODA Filters with the JSON connector

Searching a date range across any of the SODA datasets that expose a date field is via a where clause.

e.g.

NY Inmates for Month of March

URL → https://data.cityofnewyork.us/resource/gqrb-77i6.json

Parameter → $where=admitted_dt%20between%20%272017-03-01T00:00:00%27%20and%20%272017-03-31T23:59:59%27

In the above example

- Where, between, and = selection clause
- admitted_dt = unique date field for the NY inmate dataset; in other datasets, the date fields will be named differently
- %20 = a space character
- %27 = single quote character

Working URL

https://data.cityofnewyork.us/resource/gqrb-77i6.json?$where=admitted_dt%20between%20%272017-03-01T00:00:00%27%20and%20%272017-03-31T23:59:59%27

[5] What happens when the JSONpath file is saved to the designer?

The JSON designer runs in a cloud service; Each instance of a JSON data source must generate a unique token. The designer uses this token to save the JSON sample data and JSON path data to cloud storage. The connector uses the token to retrieve the JSON path file from the cloud service so it is stored locally.

Using the data source editor, the user can reopen the JSON designer and the designer will reload the sample JSON and the edited keys that constitute the JSON path file.

[6] What is the time unit used in many scientific JSON data sets?

Unix time (also known as POSIX time or epoch time) is a system for describing instants in time, defined as the number of seconds that have elapsed since 00:00:00 Coordinated Universal Time (UTC), Thursday, 1 January 1970

In the JSON designer, the user may select DATE type and lowercase “x” to specify an EPOCH time conversion

2004-09-16T23:59:58.75 = 1095379198.75

[7] Does the connector support Schema JSON format data sets?

No, the JSON connector is tuned to process record based JSON datasets only.

In the following example

https://data.ny.gov/api/views/cb42-qumz/rows.json?accessType=DOWNLOAD

This JSON (schema) dataset separates the data from the field definitions and is designed that way to allow programs to define database table schema and then populate that schema with the data found in the DATA block.

The Yellowfin JSON connector works only with record based JSON.

Like this example

https://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/4.5_week.geojson

[9] What is best practice for setting up JSON file FTP streaming?

We recommend, that you

- Use a fast FTP server as local as possible to your Yellowfin instance to ensure the best performance. The processing speed will be governed by how fast the FTP server can serve files over the LAN/WAN
- Start with an empty folder
- Copy 5 starter files to the folder
- Ensure only one type of JSON file is in that folder
- Setup your connector to access JSON formatted files from that target folder
- Choose either FTP or sFTP session
- Choose either a timestamp or deletion strategy for file handling
- Use a representative JSON file in your designer (if there are variations in your JSON files, then choose a representative that has all the keys and JSON nests; so that the designer can identify all the required keys/paths
- Select the keys required and set the correct TYPES
- Save JSONPath file to the designer and then copy back to the connector
- Set the limit field to the number of file to be processed with each report run. e.g. 50/100/200/500 (any whole integer greater than zero)
- Build your view and your first table/chart
- Confirm your table/chart represents the 5 starter files.
- Save your report and set the report scheduler frequency.
- Now you can start sending files into that folder for processing

This topic is also covered in video 2 (embedded in the connector)

[10] in REST mode, what is the purpose of the HEADER LIST field?

Some REST services are a more technical setup and the user must include header elements in each call.

Eg. Connecting the JSON connector to a single Facebook endpoint …

```
GET /v2.3/me HTTP/1.1
Host: graph.facebook.com
Authorization: <my_access_token>
Cache-Control: no-cache
Postman-Token: <postman_token>
```

So, to enter the above field in the Headers Field we concatenate and delimit by semicolon

See example below

GET /v2.3/me HTTP/1.1;Host: graph.facebook.com;Authorization: <my_access_token>;Cache-Control: no-cache;Postman-Token: <postman_token>

[11] How does the connector handle nested JSON files?

For converting nested JSON data to a structured form, we parse the JSON data recursively. Recursive parsing helps in retaining the relationships between parent and child objects and variables.

On the left below we can see a row JSON document which contains an JSON object and JSON array in the parent node and on the right, we can see the structure format of the JSON document.

Here the key name is repeated for all the rows as it is an object in the parent JSON object.

Basic Nesting

```
{
  "Name": "L_json",
  "stats": [
    {
      "age": "30",
      "marks": [21,22,23]
    },
    {
      "age": "28",
      "marks": [24,22,26]
    },
    {
      "age": "32",
      "marks": [30,22,40]
    }
  ]
}
```

Name	Age	Marks1	Marks2	Marks3
L_json	30	21	22	23
L_json	28	24	22	26
L_json	32	30	22	40

In the example below Name is repeated 4 times since it is in the parent object. Age and marks are repeated two times since they are inner parent object and score is not repeated at all as they are the children's.

So, the number of rows depends upon the no children nodes.

Complex nesting

```
{
 "Name": "L_json",
 "stats": [
  {
   "age": "30",
   "marks": "22",
   "features": [
    {
     "score": "A"
    },
    {
     "score": "B"
    }
   ]
  },
  {
   "age": "28",
   "marks": "20",
   "features": [
    {
     "score": "C"
    },
    {
     "score": "D"
    }
   ]
  }
 ]
}
```

Name	Age	Marks	Score
L_json	30	22	A
L_json	30	22	B
L_json	28	20	C
L_json	28	20	D

About Yellowfin

Yellowfin is a global Business Intelligence (BI) and analytics software vendor passionate about making BI easy. Founded in 2003 in response to the complexity and costs associated with implementing and using traditional BI tools, Yellowfin is a highly intuitive 100 percent Web-based reporting and analytics solution. Yellowfin is a leader in mobile, collaborative and embedded BI, as well as Location Intelligence and data visualization.

Over 10,000 organizations, and more than 2 million end-users across 70 different countries, use Yellowfin every day. For more information, visit www.yellowfinbi.com

Additional Resources

Yellowfin Website

Yellowfin Marketplace

Yellowfin Wiki

Support Forum

YouTube Channel