

An illustration on a blue background. A person with purple hair, wearing a light purple long-sleeved shirt and dark purple pants, stands with their back to the viewer. They are reaching up with their right hand towards a large, glowing blue sphere. To the right of the sphere is a complex network graph with white nodes and lines. The background is filled with faint, light blue network graphs and circles of various sizes.

Intersection of Knowledge Graphs and HTML-deployed Digital Twins

Demonstrated using Apple Product Pages

Situation Analysis

Increasingly, organizations are embedding domain-specific Knowledge Graphs in their web pages as a new technique in search engine optimization (SEO). For instance, Apple uses this approach across the product information and shopping action aspects of its website.



Why is this important?

Knowledge Graphs embedded within web pages typically provide a close digital rendition (a/k/a Digital Twin) of content that aids search engine computability and discoverability, while also serving the needs of a new generation of **smart agents** and **superapps** equipped with an ability to process **entity relationships** and **relationship type semantics**.

Net effect:

Better return on content management-related investment per embedded Knowledge Graph.

How does this work?

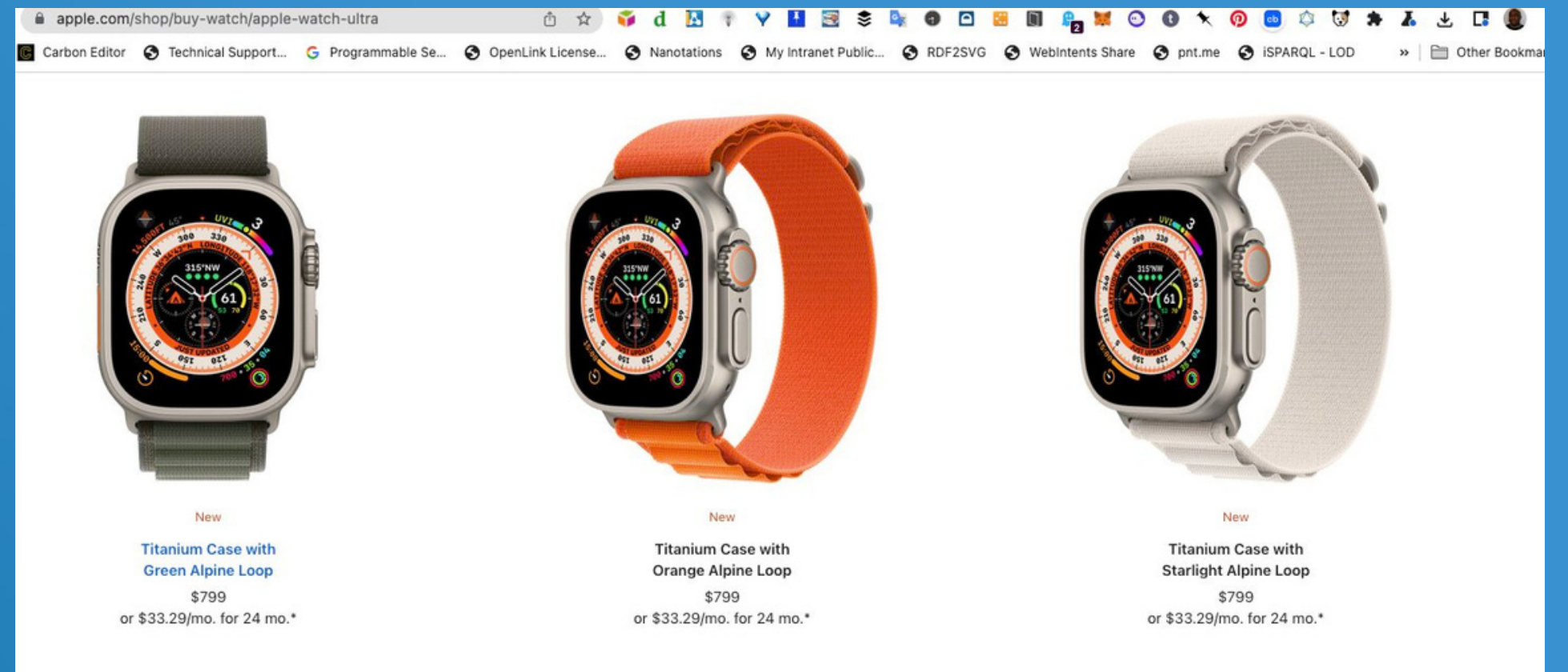
Based on sample pages from Apple's website, the slides that follow will demonstrate some new Knowledge Graph discovery and interaction capabilities that are now possible.

For instance, you can use **HTTP-based** query language & wire-protocol hybrids — such as **GraphQL** and **SPARQL** — for direct query operations.

Conventional Page View

Here we see an Apple product page about a selection of watches.

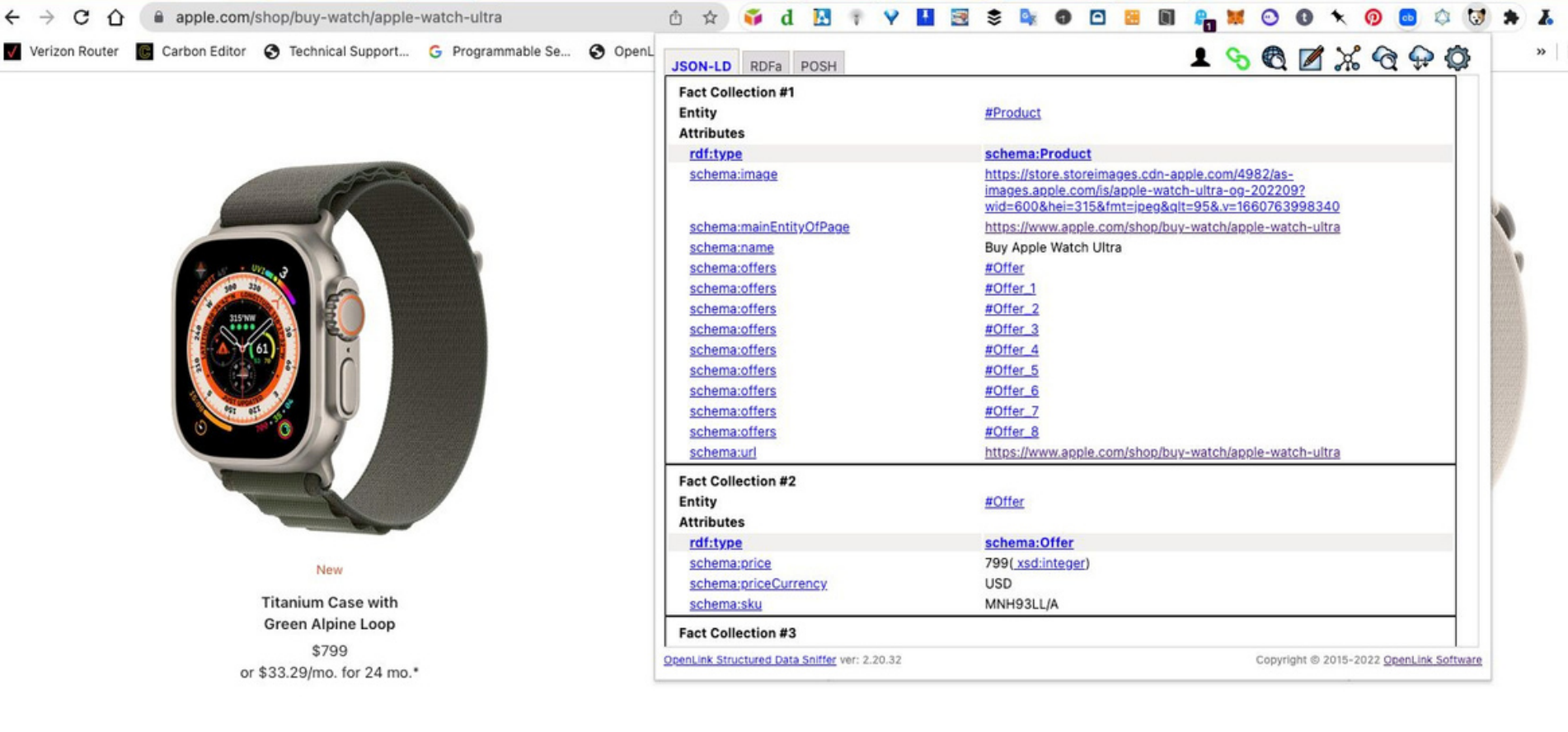
This aspect of the page is oriented towards the needs of a conventional human reader.



Embedded Product Knowledge Graph View (1)

Recently, Apple announced new Apple Watch models

Impressively, at the same time, they published web pages comprising a Product Knowledge Graph that's both human-readable and machine-computable.



The screenshot shows a web browser displaying the Apple Watch Ultra product page. On the left, there is a product image of the watch with a titanium case and a green alpine loop. Below the image, the text reads: "New Titanium Case with Green Alpine Loop \$799 or \$33.29/mo. for 24 mo.*". On the right, an embedded knowledge graph view is visible, showing three fact collections. The first collection is for the product, the second is for an offer, and the third is for another offer. The graph view includes attributes like schema:price, schema:priceCurrency, and schema:sku.

Fact Collection #1	
Entity	#Product
Attributes	
rdf:type	schema:Product
schema:image	https://store.storeimages.cdn-apple.com/4982/as-images.apple.com/is/apple-watch-ultra-09-202209?wid=600&hei=315&fmt=jpeg&qlt=95&v=1660763998340
schema:mainEntityOfPage	https://www.apple.com/shop/buy-watch/apple-watch-ultra
schema:name	Buy Apple Watch Ultra
schema:offers	#Offer
schema:offers	#Offer_1
schema:offers	#Offer_2
schema:offers	#Offer_3
schema:offers	#Offer_4
schema:offers	#Offer_5
schema:offers	#Offer_6
schema:offers	#Offer_7
schema:offers	#Offer_8
schema:url	https://www.apple.com/shop/buy-watch/apple-watch-ultra

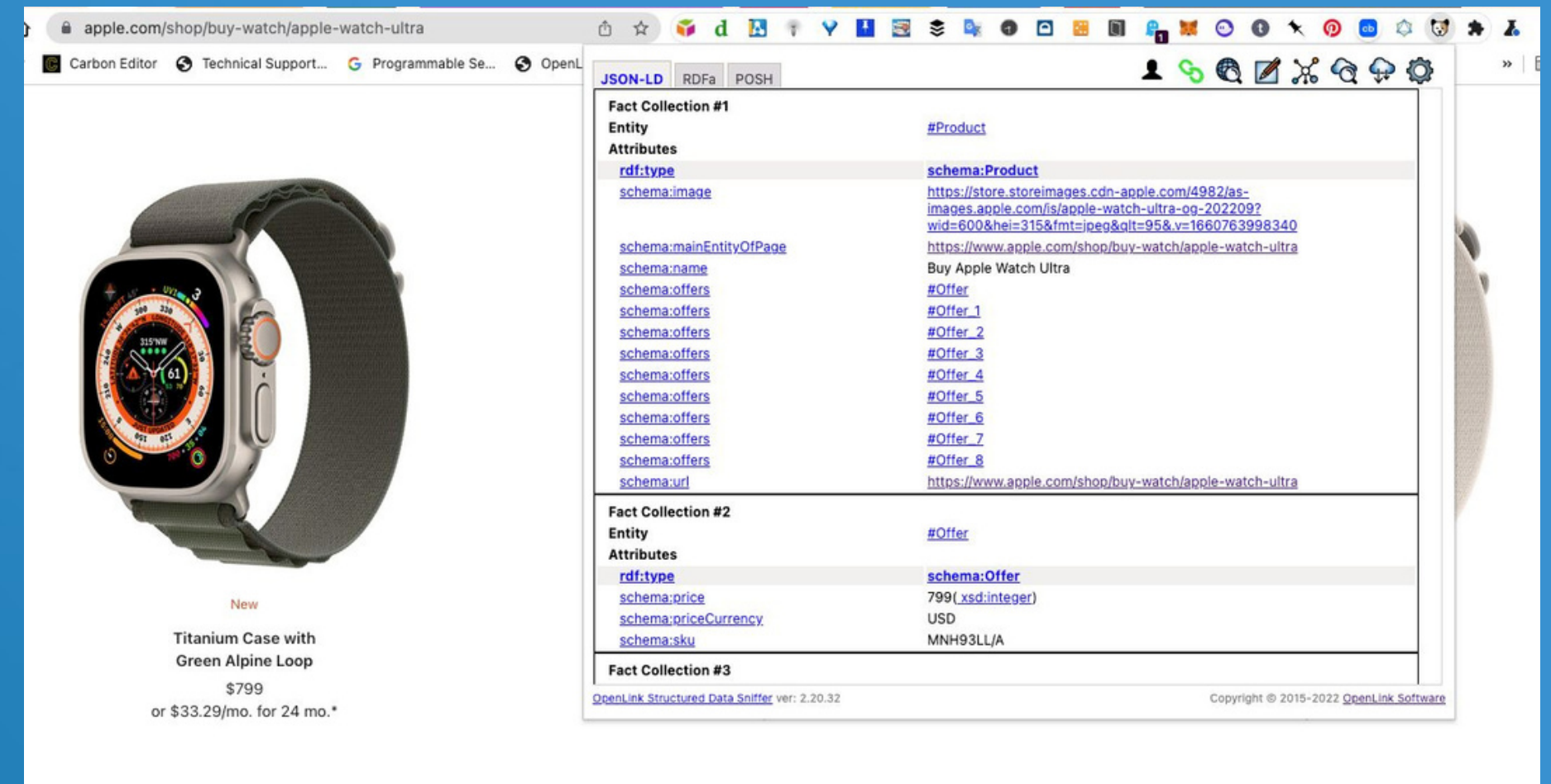
Fact Collection #2	
Entity	#Offer
Attributes	
rdf:type	schema:Offer
schema:price	799 (xsd:integer)
schema:priceCurrency	USD
schema:sku	MNH93LL/A

Fact Collection #3	
Entity	#Offer
Attributes	
rdf:type	schema:Offer
schema:price	799 (xsd:integer)
schema:priceCurrency	USD
schema:sku	MNH93LL/A

Embedded Product Knowledge Graph View (2)

How is the embedded Knowledge Graph revealed? Simply perform the following steps:

1. Install the OpenLink Structured Data Sniffer (OSDS) browser extension
2. Click the Sniffer icon 🐱 in your browser toolbar to reveal a window with tabs exposing the metadata (JSON, RDFa, etc.)



The screenshot shows the Apple Watch Ultra product page on apple.com. The OpenLink Structured Data Sniffer (OSDS) browser extension is open, displaying JSON-LD metadata for the product and offers. The metadata is organized into three fact collections:


Fact Collection #1	Entity	Value
Entity	#Product	
Attributes		
rdf:type	schema:Product	
schema:image	https://store.storeimages.cdn-apple.com/4982/as-images.apple.com/is/apple-watch-ultra-og-202209?wid=600&hei=315&fmt=jpeg&qlt=95&v=1660763998340	
schema:mainEntityOfPage	https://www.apple.com/shop/buy-watch/apple-watch-ultra	
schema:name	Buy Apple Watch Ultra	
schema:offers	#Offer	
schema:offers	#Offer_1	
schema:offers	#Offer_2	
schema:offers	#Offer_3	
schema:offers	#Offer_4	
schema:offers	#Offer_5	
schema:offers	#Offer_6	
schema:offers	#Offer_7	
schema:offers	#Offer_8	
schema:url	https://www.apple.com/shop/buy-watch/apple-watch-ultra	

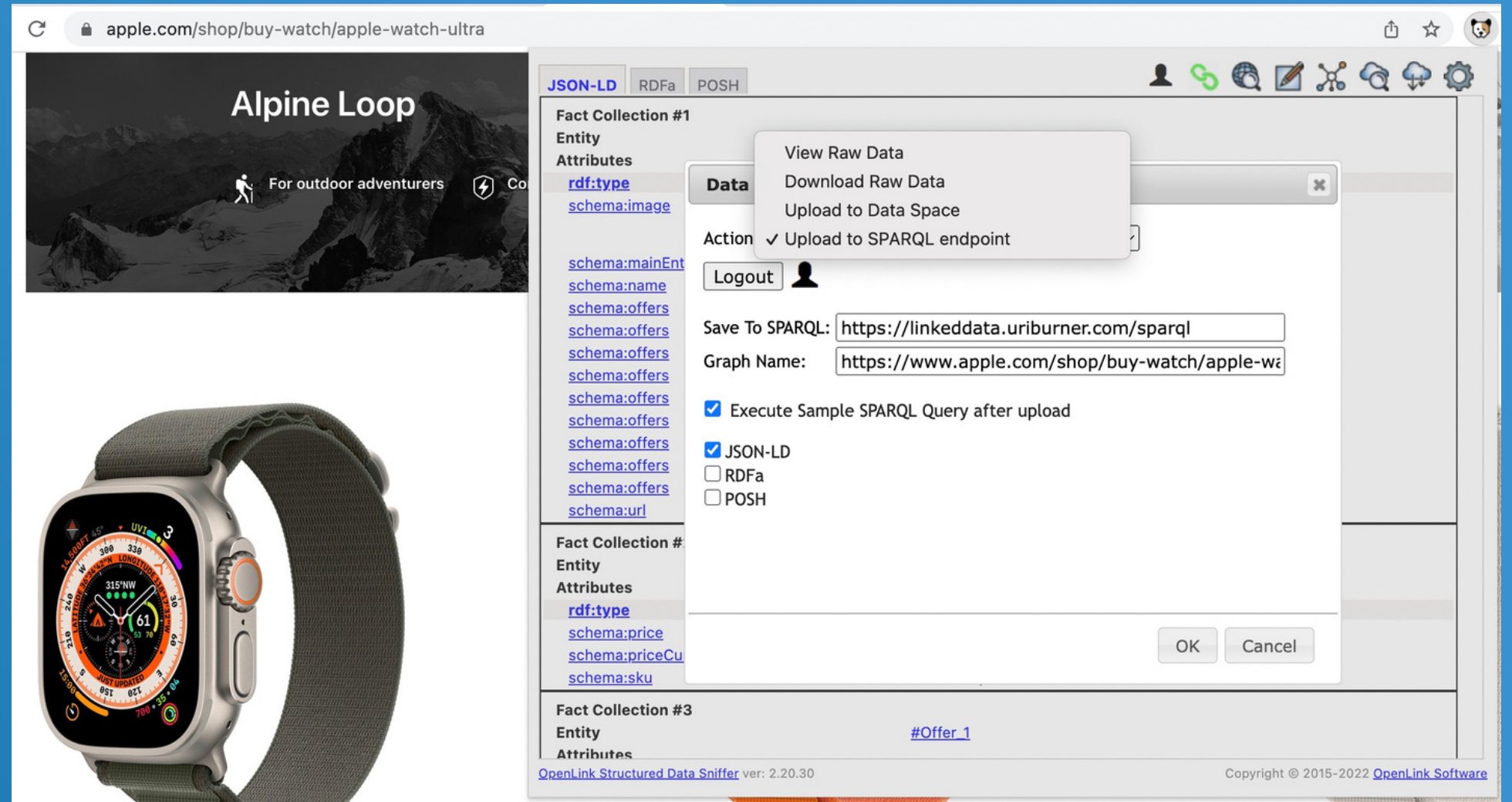
Fact Collection #2	Entity	Value
Entity	#Offer	
Attributes		
rdf:type	schema:Offer	
schema:price	799 (xsd:integer)	
schema:priceCurrency	USD	
schema:sku	MNH93LL/A	

OpenLink Structured Data Sniffer ver: 2.20.32 Copyright © 2015-2022 OpenLink Software

Load Knowledge Graph into a Database Management System (DBMS)

Using OSDS, you can also generate a SPARQL query results page using the host page URL as its Data Source Name (DSN).

How? Simply click  and pick the "Upload to SPARQL" option from the OSDS user interface!



The screenshot displays the OSDS user interface. On the left, a browser window shows the Apple website page for the 'Alpine Loop' watch. The right pane shows the OSDS interface with a 'Data' menu open, highlighting the 'Upload to SPARQL endpoint' option. The interface includes fields for 'Save To SPARQL' (https://linkeddata.uriburner.com/sparql) and 'Graph Name' (https://www.apple.com/shop/buy-watch/apple-wa). Checkboxes for 'Execute Sample SPARQL Query after upload' and 'JSON-LD' are selected. The interface also shows a list of 'Fact Collection' items with their respective entities and attributes.

Knowledge Graph Introspection & Exploration (1)

Following data upload, you will be presented with a Knowledge Graph introspection & exploration page, constructed from a SPARQL query.

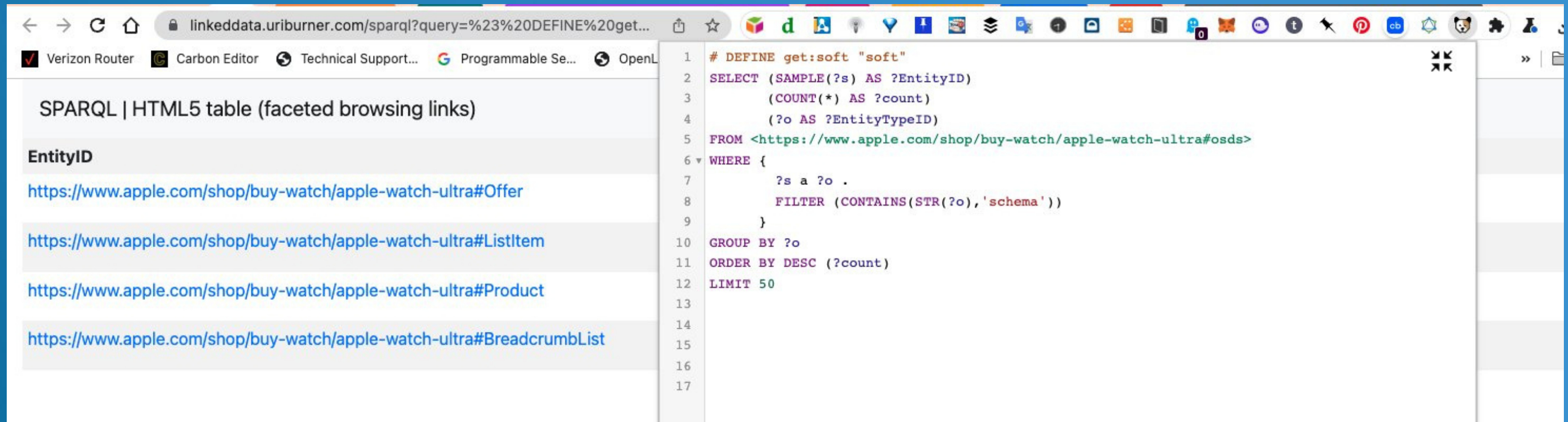
Here's a live example of the [SPARQL results page](#) which uses the Apple Watch page URL as its Data Source Name (DSN)

SPARQL | HTML5 table (faceted browsing links)

EntityID	count	EntityTypeID
https://www.apple.com/shop/buy-watch/apple-watch-ultra#Offer	9	http://schema.org/Offer
https://www.apple.com/shop/buy-watch/apple-watch-ultra#ListItem	2	http://schema.org/ListItem
https://www.apple.com/shop/buy-watch/apple-watch-ultra#Product	1	http://schema.org/Product
https://www.apple.com/shop/buy-watch/apple-watch-ultra#BreadcrumbList	1	http://schema.org/BreadcrumbList

Knowledge Graph Introspection & Exploration (2)

You can also edit this query [SPARQL query result page](#) inline using OSDS, as shown in the image below.



The screenshot shows a web browser window with a SPARQL query editor and its results. The browser's address bar shows the URL: `linkeddata.uriburner.com/sparql?query=%23%20DEFINE%20get...`. The page title is "SPARQL | HTML5 table (faceted browsing links)". The results are displayed in a table with the following content:

EntityID
https://www.apple.com/shop/buy-watch/apple-watch-ultra#Offer
https://www.apple.com/shop/buy-watch/apple-watch-ultra#ListItem
https://www.apple.com/shop/buy-watch/apple-watch-ultra#Product
https://www.apple.com/shop/buy-watch/apple-watch-ultra#BreadcrumbList

The query editor on the right shows the following SPARQL query:

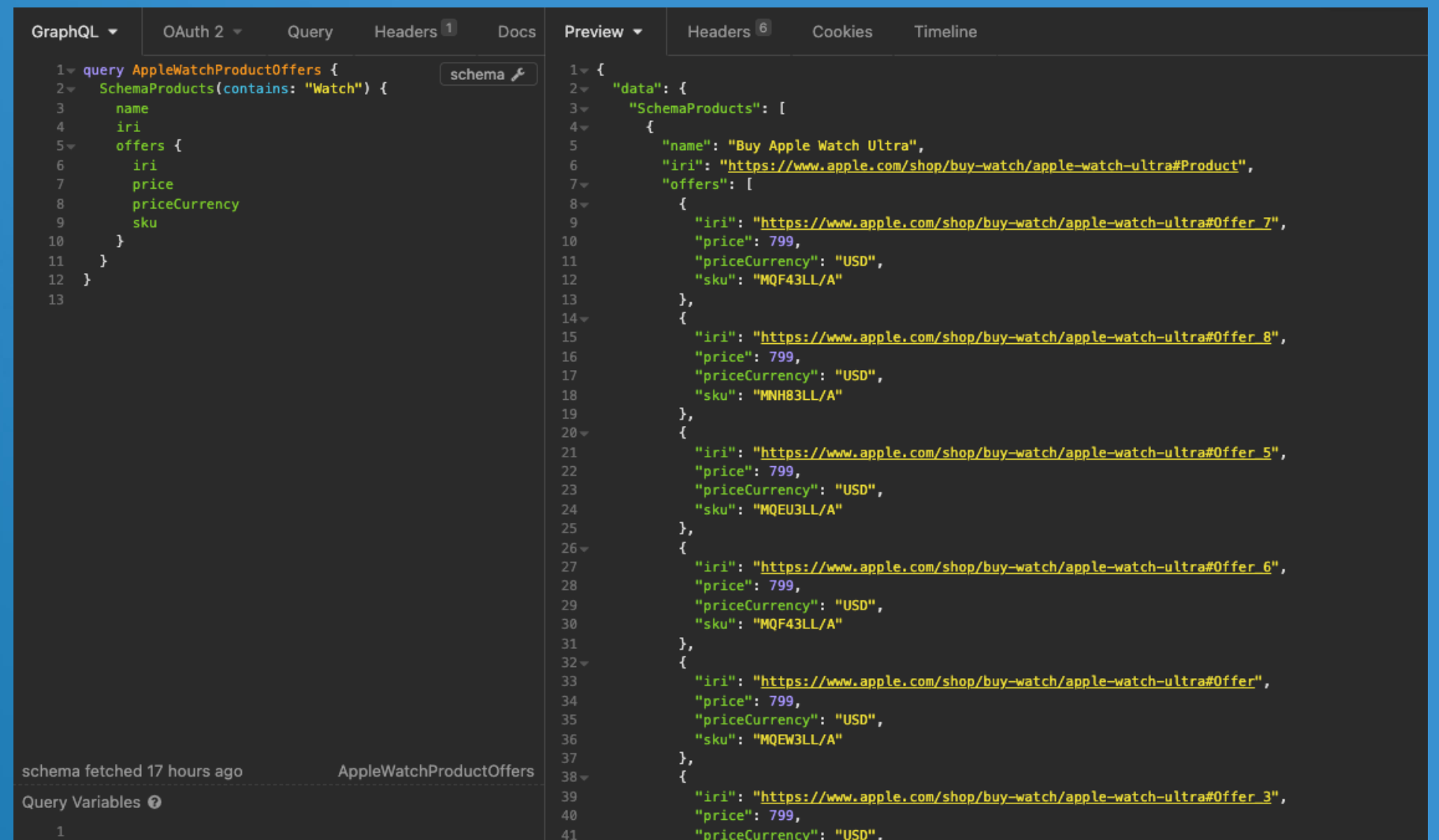
```
1 # DEFINE get:soft "soft"
2 SELECT (SAMPLE(?s) AS ?EntityID)
3         (COUNT(*) AS ?count)
4         (?o AS ?EntityTypeID)
5 FROM <https://www.apple.com/shop/buy-watch/apple-watch-ultra#osds>
6 WHERE {
7     ?s a ?o .
8     FILTER (CONTAINS(STR(?o), 'schema'))
9 }
10 GROUP BY ?o
11 ORDER BY DESC (?count)
12 LIMIT 50
13
14
15
16
17
```

GraphQL Interaction with Knowledge Graph

You can also use GraphQL to query the same Knowledge Graph

Example:

- [GraphQL query](#) targeting data from the same Apple Watch page
- [GraphQL query results page](#)



The screenshot displays a GraphQL IDE interface. On the left, the query editor shows a query named 'AppleWatchProductOffers' that filters products containing 'Watch' and lists their offers with fields for name, IRI, price, price currency, and SKU. On the right, the preview pane shows the JSON response, which is a list of offers for the 'Buy Apple Watch Ultra' product, including their IRIs, prices (799 USD), and SKUs.

```
1 query AppleWatchProductOffers {
2   SchemaProducts(contains: "Watch") {
3     name
4     iri
5     offers {
6       iri
7       price
8       priceCurrency
9       sku
10    }
11  }
12 }
13
```

```
1 {
2   "data": {
3     "SchemaProducts": [
4       {
5         "name": "Buy Apple Watch Ultra",
6         "iri": "https://www.apple.com/shop/buy-watch/apple-watch-ultra#Product",
7         "offers": [
8           {
9             "iri": "https://www.apple.com/shop/buy-watch/apple-watch-ultra#offer 7",
10            "price": 799,
11            "priceCurrency": "USD",
12            "sku": "MQF43LL/A"
13          },
14          {
15            "iri": "https://www.apple.com/shop/buy-watch/apple-watch-ultra#offer 8",
16            "price": 799,
17            "priceCurrency": "USD",
18            "sku": "MNH83LL/A"
19          },
20          {
21            "iri": "https://www.apple.com/shop/buy-watch/apple-watch-ultra#offer 5",
22            "price": 799,
23            "priceCurrency": "USD",
24            "sku": "MQEU3LL/A"
25          },
26          {
27            "iri": "https://www.apple.com/shop/buy-watch/apple-watch-ultra#offer 6",
28            "price": 799,
29            "priceCurrency": "USD",
30            "sku": "MQF43LL/A"
31          },
32          {
33            "iri": "https://www.apple.com/shop/buy-watch/apple-watch-ultra#offer",
34            "price": 799,
35            "priceCurrency": "USD",
36            "sku": "MQEW3LL/A"
37          },
38          {
39            "iri": "https://www.apple.com/shop/buy-watch/apple-watch-ultra#offer 3",
40            "price": 799,
41            "priceCurrency": "USD",
```

Net Result

Web Pages are a new structured data source, just like relational tables in a SQL-accessible DBMS, for Knowledge Graph discovery and interaction.

As demonstrated in this presentation, our tools aid organizations and individuals alike in their pursuit of agility, driven by productive interaction with data — leveraging new frontiers enabled by a new generation of data access protocols and query languages.



Conclusion

The growing practice of deploying Knowledge Graphs via web pages offers a cost-effective and practical solution to several challenges:

- Conventional Search Engine Optimization (SEO)
- Emerging Semantic SEO (SSEO) — where machine-computable Expertise, Authority, and Trust (E-A-T) becomes a vital factor in Search Engine algorithms
- Data Access, Integration, Virtualization, and Management — where declarative query language interactions are extended beyond conventional RDBMS tables

How do I get started — Client Side?

Simply install the OpenLink Structured Data Sniffer (OSDS) extension in your browser and then visit a page of interest. Upon page display, the OSDS icon will be visually activated, indicating metadata discovery. When you click the OSDS icon, you will be presented with a visualization of transformed metadata.

OSDS Download options:

- [Chrome Store](#)
- [Mozilla Firefox Store](#)
- [Apple App Store](#)
- [GitHub project page](#)

How do I get started — Server Side?

Simply install our Virtuoso multi-model DBMS and Knowledge Graph deployment platform using any of the following options:

- [Our Live URIBurner Service](#) — comprising a progressively enhanced Knowledge Graph
- [Your own Virtuoso Enterprise Edition instance](#) running on macOS, Linux, or Windows
- [Your own Docker Container](#)
- [Your own Amazon AWS Cloud instance](#) — BYOL or PAGO
- [Your own Microsoft Azure Cloud instance](#) — BYOL or PAGO
- Your own Virtuoso Open Source Edition instance, built or downloaded from [our GitHub project repository](#).



ADDITIONAL INFORMATION

- [What is the OpenLink Structured Data Sniffer, and Why is it Important?](#)
- [OpenLink Software Home Page](#)
- [OpenLink Community Forum](#)
- [Linked Data Ontology and Knowledge Graph Explainer](#)

