Developer Days
Automation

The bridge to possible

Crosswork Hierarchical Controller

The API to your network

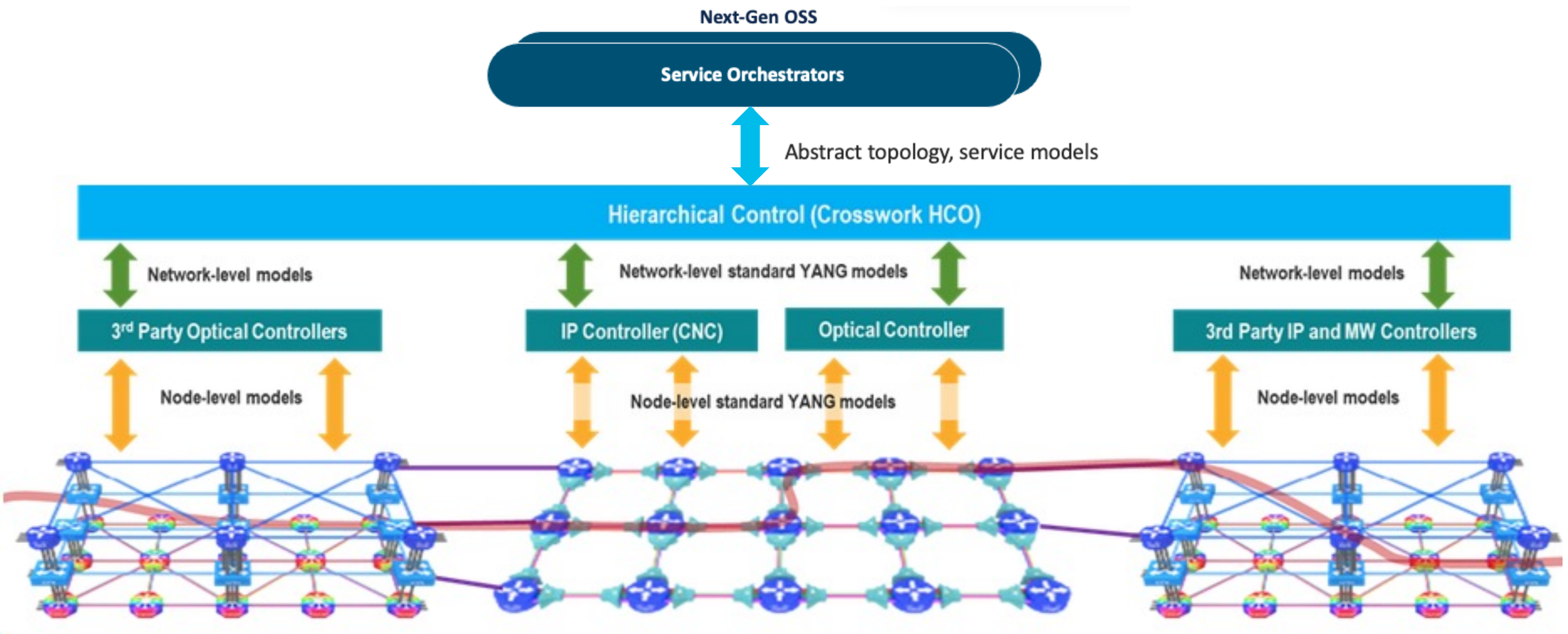
Yona Shikhmanter – Customer Success Product Manager
Daniel Kraus – CX Architect
May 11th 2023

Agenda

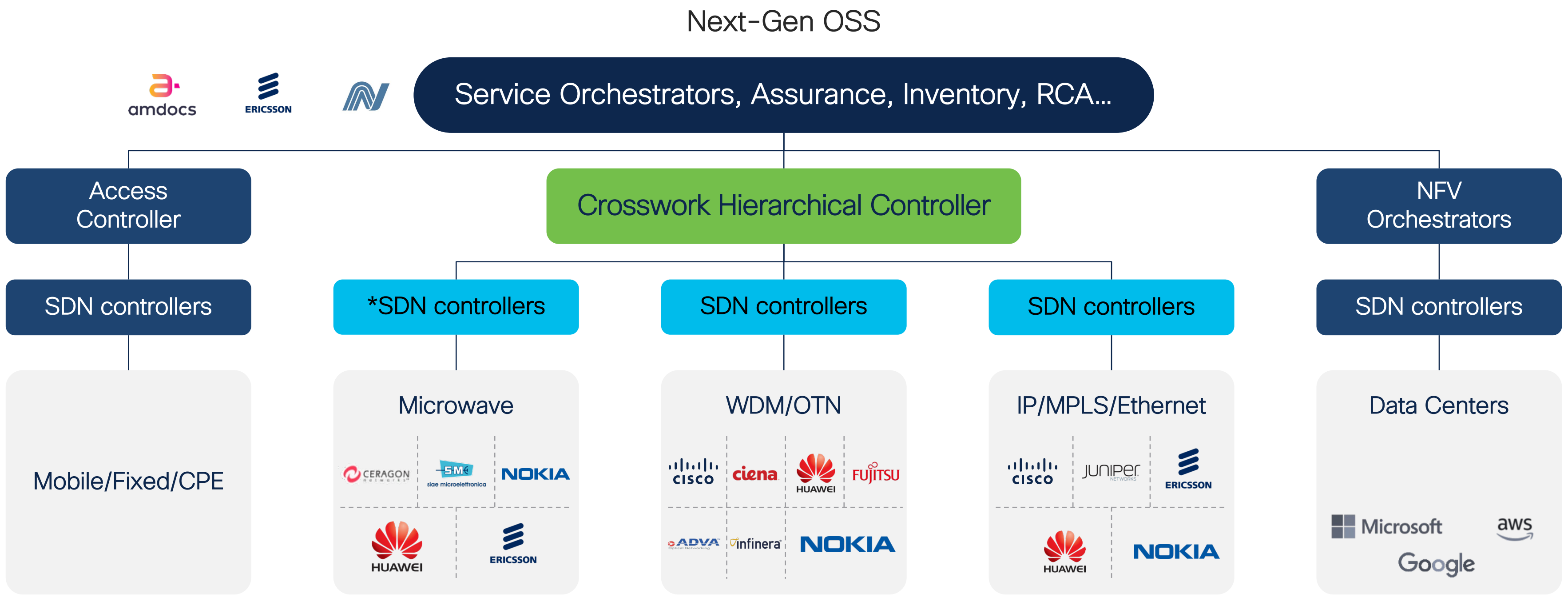
- Introduction to CHCO
  - High-level overview
  - First glance at CHCO UI
- SHQL from 0 to hero
- SHQL demo and examples
  - SHQL App
  - REST API
  - CLI
  - NSO

What is (C)HCO?

CHCO – Crosswork Hierarchical Controller

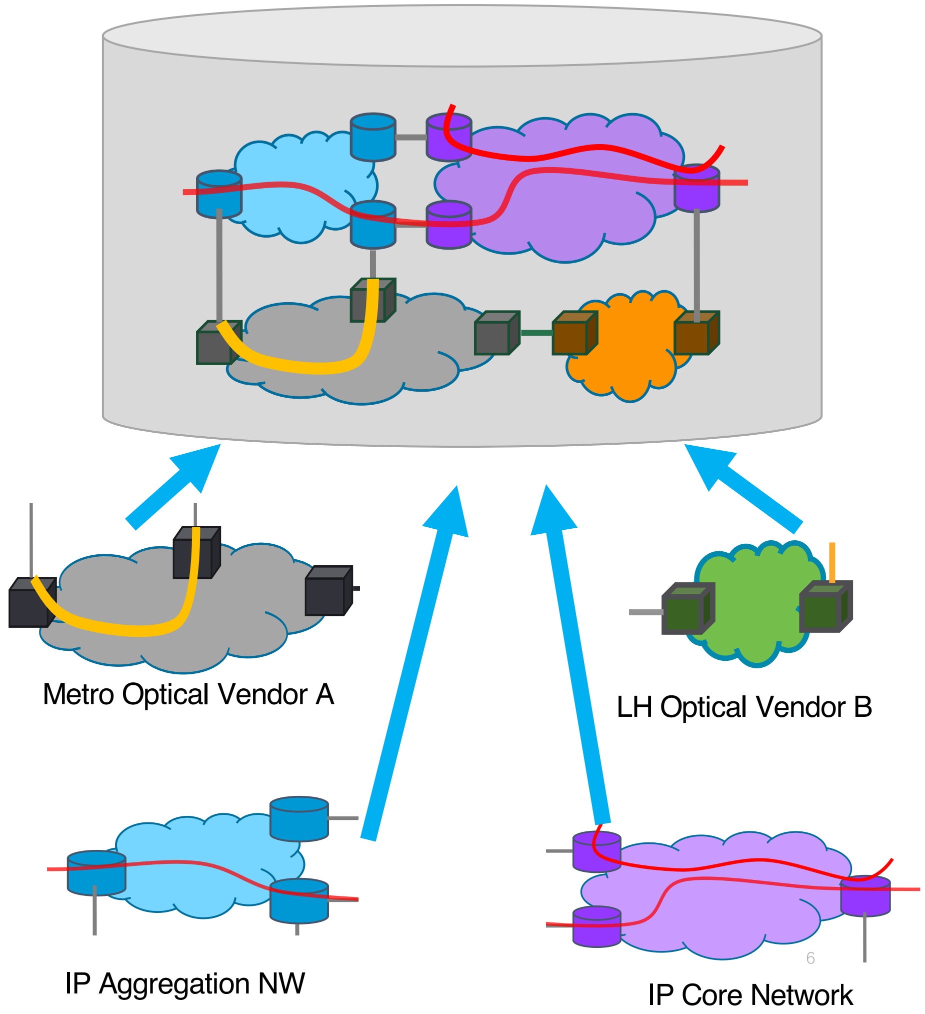

Next-Gen OSS
Service Orchestrators
Abstract topology, service models
Hierarchical Control (Crosswork HCO)
Network-level models
Network-level standard YANG models
Network-level models
3rd Party Optical Controllers
IP Controller (CNC)
Optical Controller
3rd Party IP and MW Controllers
Node-level models
Node-level standard YANG models
Node-level models

CHCO – Crosswork Hierarchical Controller

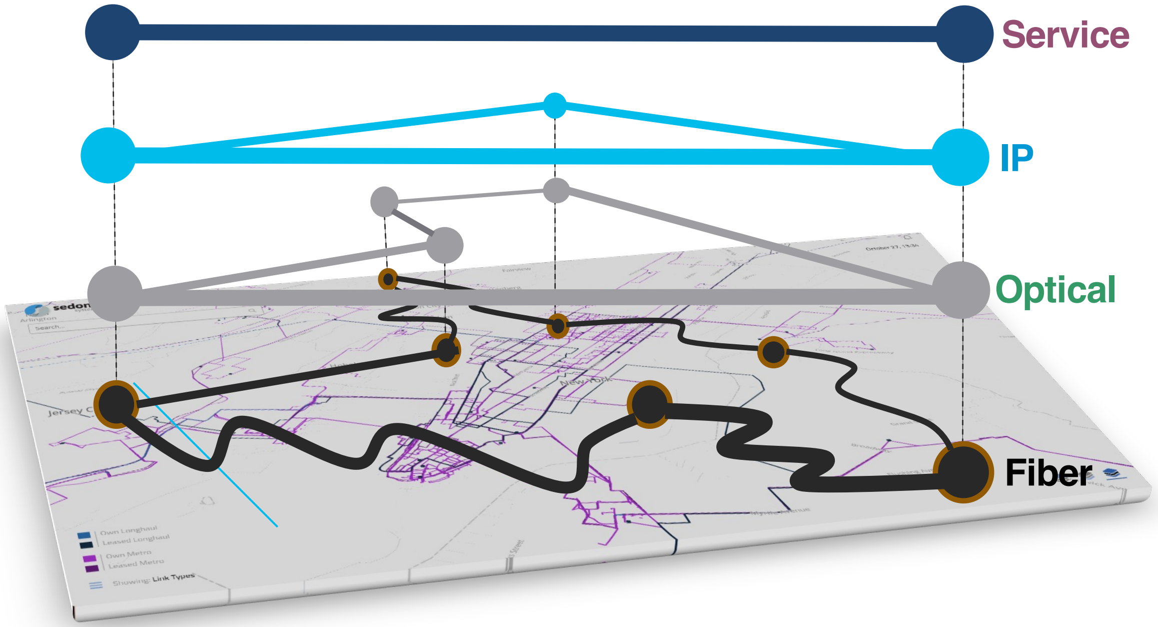


Crosswork Hierarchical Controller pre-integrated with most of the vendors

Putting the Network Puzzle Together



From Fiber to Service

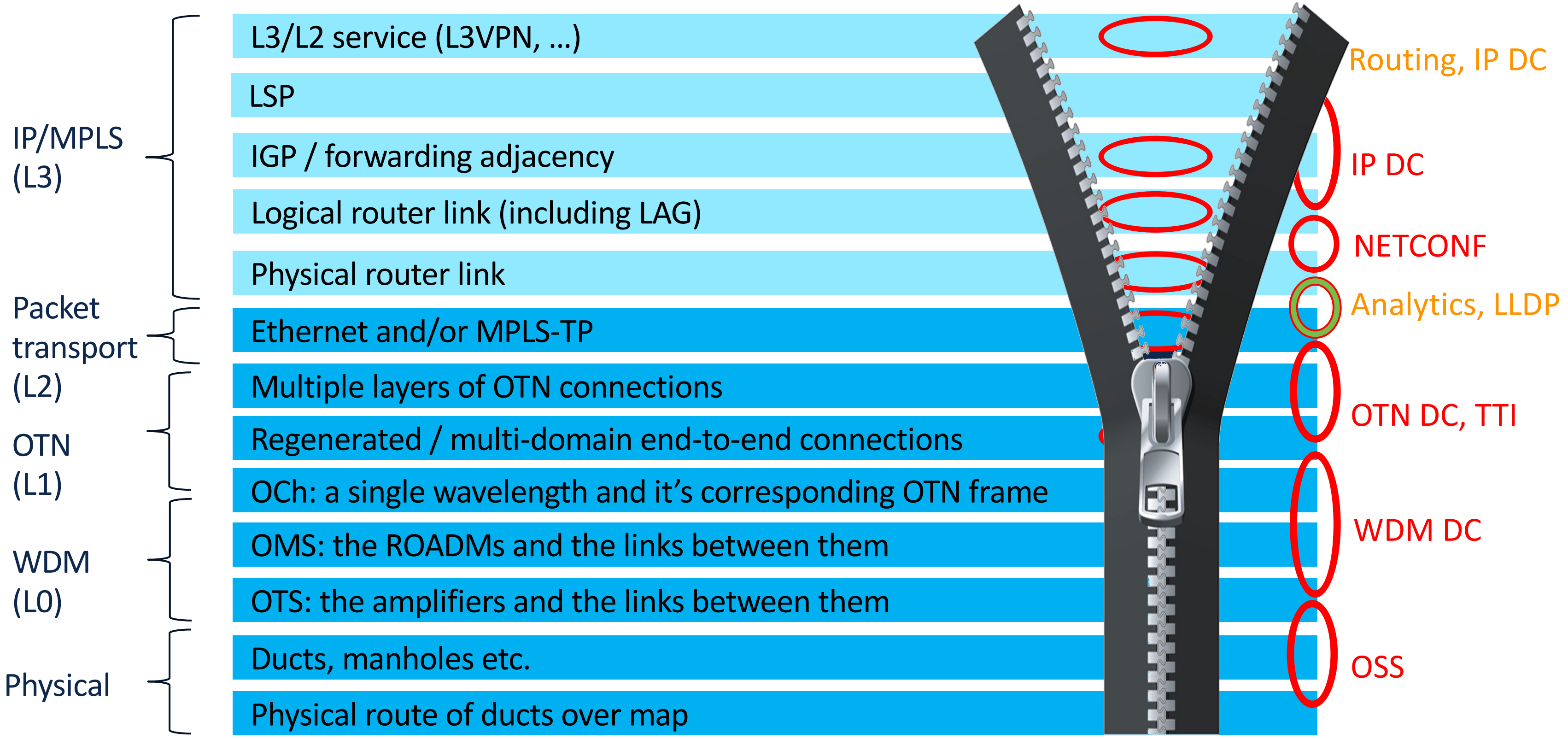


Complete: multi-layer, multi-vendor & multi-domain topology, traffic and services (SDN & legacy)

Current: automatically & ongoingly discovered – directly from the network

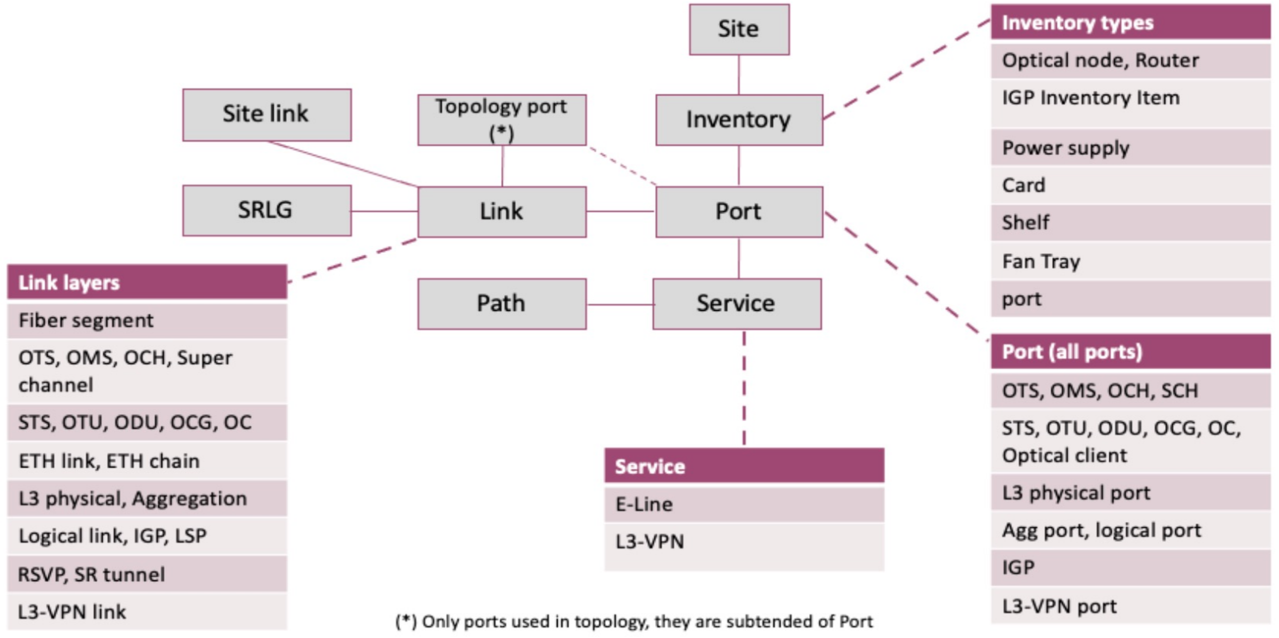
Correlated: dynamically deducing cross-domain connectivity

The CHCO model



- L3/L2 service (L3VPN, …)
- LSP
- IGP / forwarding adjacency
- Logical router link (including LAG)
- Physical router link
- Ethernet and/or MPLS-TP
- Multiple layers of OTN connections
- Regenerated / multi-domain end-to-end connections
- OCh: a single wavelength and it's corresponding OTN frame
- OMS: the ROADMs and the links between them
- OTS: the amplifiers and the links between them
- Ducts, manholes etc.
- Physical route of ducts over map

The CHCO model



Inventory types
Optical node, Router
IGP Inventory Item
Power supply
Card
Shelf
Fan Tray
port

Link layers
Fiber segment
OTS, OMS, OCH, Super channel
STS, OTU, ODU, OCG, OC
ETH link, ETH chain
L3 physical, Aggregation
Logical link, IGP, LSP
RSVP, SR tunnel
L3-VPN link

Service
E-Line
L3-VPN

Port (all ports)
OTS, OMS, OCH, SCH
STS, OTU, ODU, OCG, OC, Optical client
L3 physical port
Agg port, logical port
IGP
L3-VPN port

(*) Only ports used in topology, they are subtended of Port

What is CHCO – A quick summary

- Hierarchical, multi-layer, multi-domain, multi-vendor
- Always up-to-date network model
- Single pane of glass
  - Provisioning
  - Visualization
  - Assurance

First glance at CHCO UI

SHQL – Sedona Hierarchical Query Language

SHQL – The network query language

- Problem statement

- Raw network data model too complex for BI tools
- Costly OSS development due to need to ingest complex network models

SHQL – The network query language

Solution

Sedona Hierarchical Query Language (SHQL)

Extract complex network data across layers, in a simple, flat structure

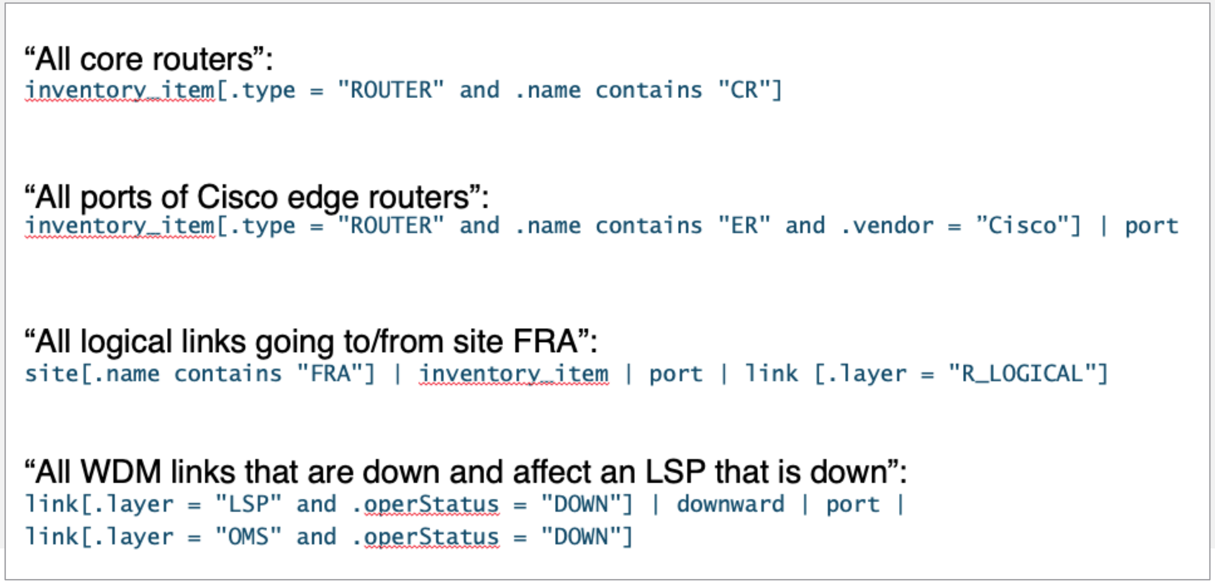
- Navigation up and down the layers
- Transform from one object type to another
- Integrated time machine

Usage patterns

- By HCO **apps**
- Creation of customized, rule-based **tags**
- Through Hierarchical Controller REST **API**
- Through CLI
- Through NSO NED (REST API)

SHQL – The network query language

TIP Examples



"All core routers":

```
inventory_item[.type = "ROUTER" and .name contains "CR"]
```

"All ports of Cisco edge routers":

```
inventory_item[.type = "ROUTER" and .name contains "ER" and .vendor = "Cisco"] | port
```

"All logical links going to/from site FRA":

```
site[.name contains "FRA"] | inventory_item | port | link [.layer = "R_LOGICAL"]
```

"All WDM links that are down and affect an LSP that is down":

```
link[.layer = "LSP" and .operStatus = "DOWN"] | downward | port |
link[.layer = "OMS" and .operStatus = "DOWN"]
```

First glance at the CHCO UI

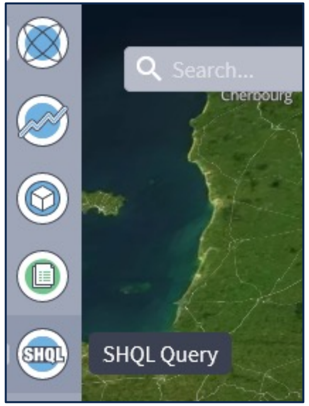
Overview

1. SHQL enables you to query the HCO model in an intuitive way
2. Replies are in JSON format
3. There are many filters, commands and options to make your query sharp to the point
4. Reply's data structure can be manipulated
5. SHQL can be queried by API

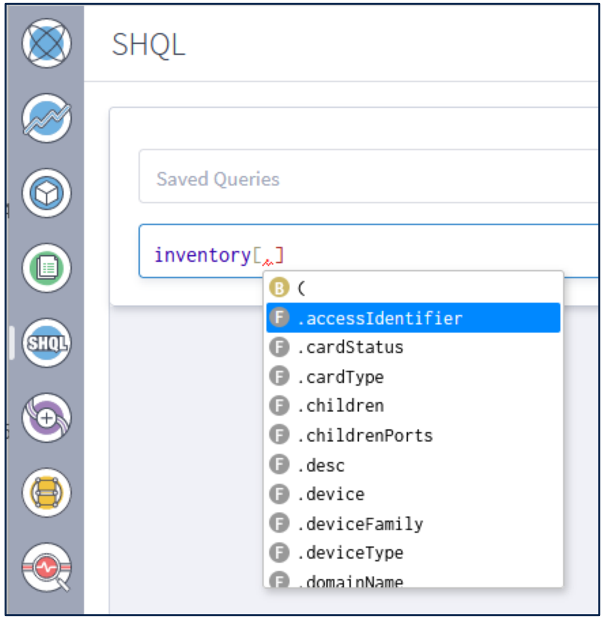
Using SHQL

SHQL Application

- An intuitive application
- Auto completion and suggestion of possible objects, attributes and operands



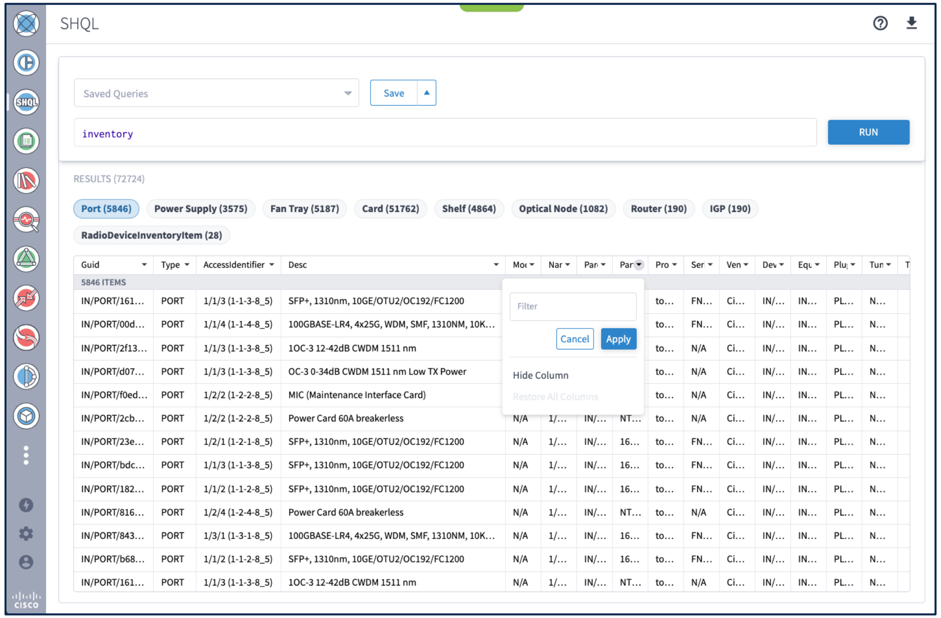
* The application's icon can be found on the left bar



```
inventory[.]
(
.accessIdentifier
.cardStatus
.cardType
.children
.childrenPorts
.desc
.device
.deviceFamily
.deviceType
.domainName
```

Using "." (dot) will display all the properties related to the selected item

- The retrieved data is organized under its related tabs by model objects.
- You can sort and filter values by column.



```
inventory
```

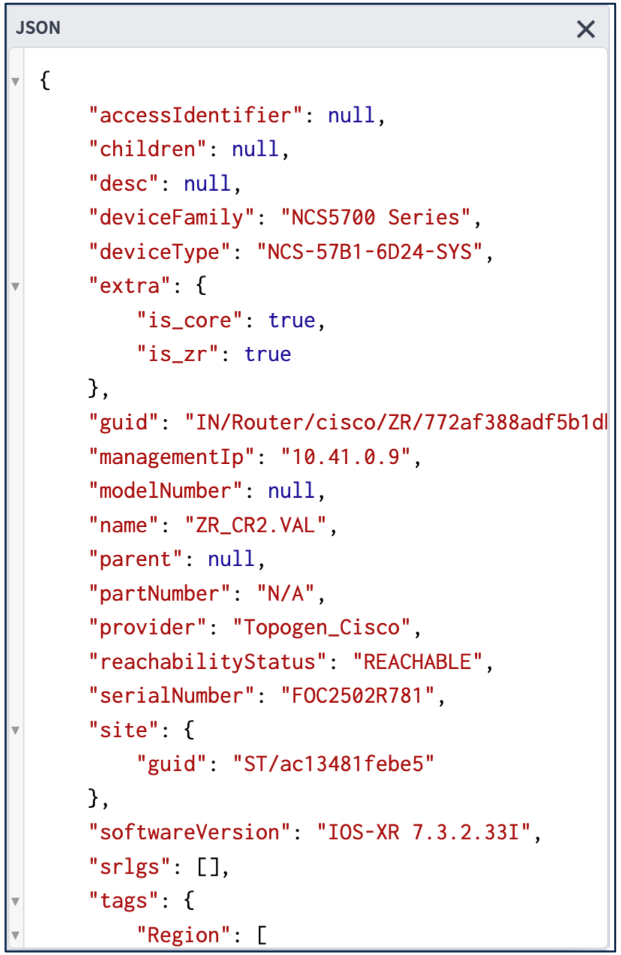
RESULTS (72724)

Port (5846) | Power Supply (3575) | Fan Tray (5187) | Card (51762) | Shelf (4864) | Optical Node (1082) | Router (190) | IGP (190) | RadioDeviceInventoryItem (28)

5846 ITEMS

Guid	Type	AccessIdentifier	Desc	Moc	Nar	Par	Par	Pro	Ser	Ven	Dev	Equ	Plu	Tun	T
IN/PORT/161…	PORT	1/1/3 (1-1-3-8_5)	SFP+, 1310nm, 10GE/OTU2/OC192/FC1200					to…	FN…	Ci…	IN/…	IN…	PL…	N…	
IN/PORT/00d…	PORT	1/1/4 (1-1-4-8_5)	100GBASE-LR4, 4x25G, WDM, SMF, 1310NM, 10K…					to…	FN…	Ci…	IN/…	IN…	PL…	N…	
IN/PORT/2f13…	PORT	1/1/3 (1-1-3-8_5)	1OC-3 12-42dB CWDM 1511 nm					to…	N/A	Ci…	IN/…	IN…	PL…	N…	
IN/PORT/d07…	PORT	1/1/3 (1-1-3-8_5)	OC-3 0-34dB CWDM 1511 nm Low TX Power					to…	N/A	Ci…	IN/…	IN…	PL…	N…	
IN/PORT/f0ed…	PORT	1/2/2 (1-2-2-8_5)	MIC (Maintenance Interface Card)					to…	N/A	Ci…	IN/…	IN…	PL…	N…	
IN/PORT/2cb…	PORT	1/2/2 (1-2-2-8_5)	Power Card 60A breakerless	N/A	1/…	IN/…	NT…	to…	N/A	Ci…	IN/…	IN…	PL…	N…	
IN/PORT/23e…	PORT	1/2/1 (1-2-1-8_5)	SFP+, 1310nm, 10GE/OTU2/OC192/FC1200	N/A	1/…	IN/…	16…	to…	FN…	Ci…	IN/…	IN…	PL…	N…	
IN/PORT/bdc…	PORT	1/1/3 (1-1-3-8_5)	SFP+, 1310nm, 10GE/OTU2/OC192/FC1200	N/A	1/…	IN/…	16…	to…	FN…	Ci…	IN/…	IN…	PL…	N…	
IN/PORT/182…	PORT	1/1/2 (1-1-2-8_5)	SFP+, 1310nm, 10GE/OTU2/OC192/FC1200	N/A	1/…	IN/…	16…	to…	FN…	Ci…	IN/…	IN…	PL…	N…	
IN/PORT/816…	PORT	1/2/4 (1-2-4-8_5)	Power Card 60A breakerless	N/A	1/…	IN/…	NT…	to…	N/A	Ci…	IN/…	IN…	PL…	N…	
IN/PORT/843…	PORT	1/3/1 (1-3-1-8_5)	100GBASE-LR4, 4x25G, WDM, SMF, 1310NM, 10K…	N/A	1/…	IN/…	16…	to…	FN…	Ci…	IN/…	IN…	PL…	N…	
IN/PORT/b68…	PORT	1/1/2 (1-1-2-8_5)	SFP+, 1310nm, 10GE/OTU2/OC192/FC1200	N/A	1/…	IN/…	16…	to…	FN…	Ci…	IN/…	IN…	PL…	N…	
IN/PORT/161…	PORT	1/1/3 (1-1-3-8_5)	1OC-3 12-42dB CWDM 1511 nm	N/A	1/…	IN/…	NT…	to…	N/A	Ci…	IN/…	IN…	PL…	N…	

Selecting object from the table, displays its properties in JSON format.


JSON

```
{
    "accessIdentifier": null,
    "children": null,
    "desc": null,
    "deviceFamily": "NCS5700 Series",
    "deviceType": "NCS-57B1-6D24-SYS",
    "extra": {
        "is_core": true,
        "is_zr": true
    },
    "guid": "IN/Router/cisco/ZR/772af388adf5b1dl
    "managementIp": "10.41.0.9",
    "modelNumber": null,
    "name": "ZR_CR2.VAL",
    "parent": null,
    "partNumber": "N/A",
    "provider": "Topogen_Cisco",
    "reachabilityStatus": "REACHABLE",
    "serialNumber": "FOC2502R781",
    "site": {
        "guid": "ST/ac13481febe5"
    },
    "softwareVersion": "IOS-XR 7.3.2.33I",
    "srlgs": [],
    "tags": {
        "Region": [
```

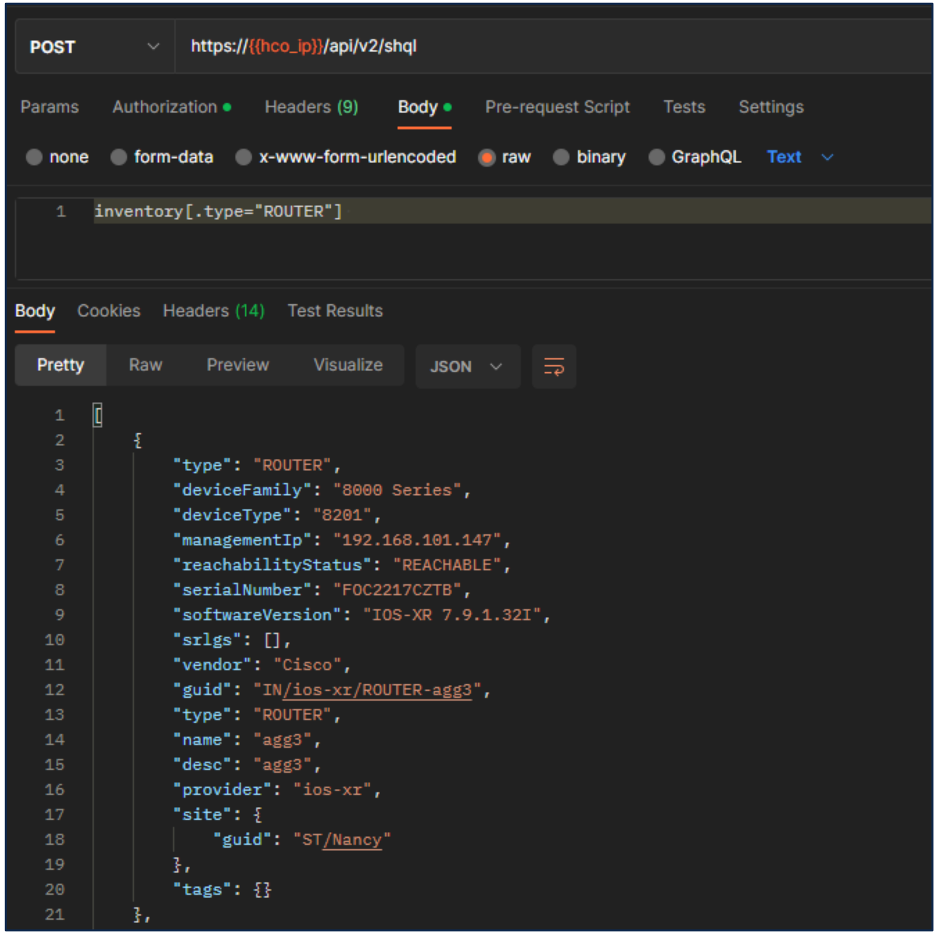
Saving\Deleting Queries

- A query can be stored in the “Saved Queries” dropdown menu for repeated use.
- To save a query, click the “Save As” button and a dialog box appears, type a name for the query and click Save.
- Select a query from the dropdown list of saved queries.
- Click “Delete Query” button, a confirmation message appears click “Delete” again and the query is removed from the list.

REST API

- SHQL queries can be sent by REST API and get results in JSON format.
- URL: https://<HCO_IP>/api/v2/shql
- Use POST method
- Add the SHQL query to the body as text

REST API

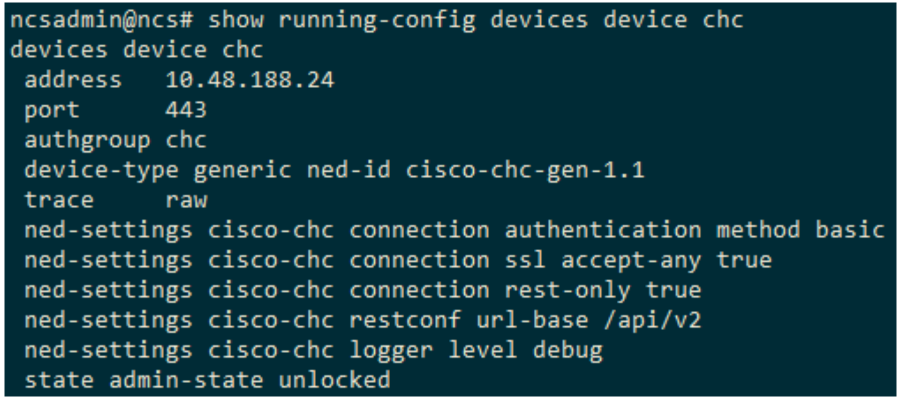


```
POST https://{{hco_ip}}/api/v2/shql
```

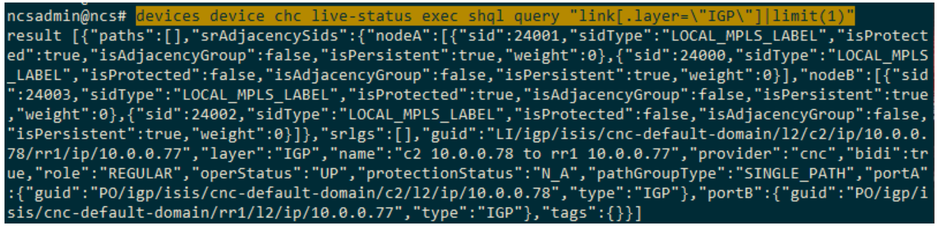
```
inventory[.type="ROUTER"]
```

```
[
    {
        "type": "ROUTER",
        "deviceFamily": "8000 Series",
        "deviceType": "8201",
        "managementIp": "192.168.101.147",
        "reachabilityStatus": "REACHABLE",
        "serialNumber": "FOC2217CZTB",
        "softwareVersion": "IOS-XR 7.9.1.32I",
        "srlgs": [],
        "vendor": "Cisco",
        "guid": "IN/ios-xr/ROUTER-agg3",
        "type": "ROUTER",
        "name": "agg3",
        "desc": "agg3",
        "provider": "ios-xr",
        "site": {
            "guid": "ST/Nancy"
        },
        "tags": {}
    },
```

NSO



```
ncsadmin@ncs# show running-config devices device chc
devices device chc
 address   10.48.188.24
 port      443
 authgroup chc
 device-type generic ned-id cisco-chc-gen-1.1
 trace     raw
 ned-settings cisco-chc connection authentication method basic
 ned-settings cisco-chc connection ssl accept-any true
 ned-settings cisco-chc connection rest-only true
 ned-settings cisco-chc restconf url-base /api/v2
 ned-settings cisco-chc logger level debug
 state admin-state unlocked
```



```
ncsadmin@ncs# devices device chc live-status exec shql query "link[.layer=\"IGP\"]|limit(1)"
result [{"paths":[],"srAdjacencySids":{"nodeA":[{"sid":24001,"sidType":"LOCAL_MPLS_LABEL","isProtect
ed":true,"isAdjacencyGroup":false,"isPersistent":true,"weight":0},{"sid":24000,"sidType":"LOCAL_MPLS
_LABEL","isProtected":false,"isAdjacencyGroup":false,"isPersistent":true,"weight":0}],"nodeB":[{"sid
":24003,"sidType":"LOCAL_MPLS_LABEL","isProtected":true,"isAdjacencyGroup":false,"isPersistent":true
,"weight":0},{"sid":24002,"sidType":"LOCAL_MPLS_LABEL","isProtected":false,"isAdjacencyGroup":false,
"isPersistent":true,"weight":0}]},"srlgs":[],"guid":"LI/igp/isis/cnc-default-domain/l2/c2/ip/10.0.0.
78/rr1/ip/10.0.0.77","layer":"IGP","name":"c2 10.0.0.78 to rr1 10.0.0.77","provider":"cnc","bidi":tr
ue,"role":"REGULAR","operStatus":"UP","protectionStatus":"N_A","pathGroupType":"SINGLE_PATH","portA"
:{"guid":"PO/igp/isis/cnc-default-domain/c2/l2/ip/10.0.0.78","type":"IGP"},"portB":{"guid":"PO/igp/i
sis/cnc-default-domain/rr1/l2/ip/10.0.0.77","type":"IGP"},"tags":{}}]
```

SHQL Conditions

Conditions

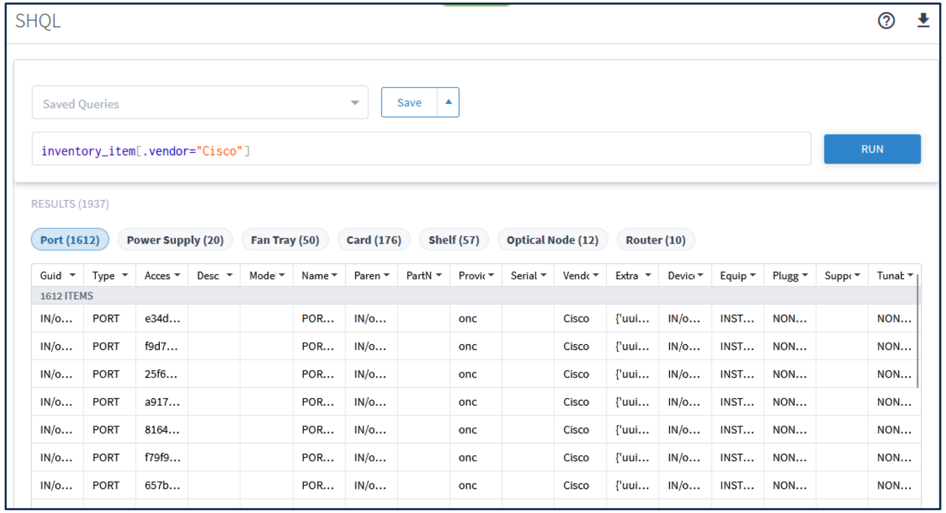
Operand	Numerical	String	Description
=	✓	✓	Equal to.
>	✓		Larger than.
>=	✓		Larger than or equal to.
contains		✓	Partial match.
endswith		✓	Ending with a given pattern.
has			Item in an array. Use to look for an item when the field is a list.
in	✓	✓	Matched list of patterns. Use when the field is a single item and the filter contains multiple items.
intersect	✓		Geographical intersection of regions at a specific longitude and latitude. For example: region[.geometry intersect (4.8945398, 52.3666)]
is		✓	Boolean (true / false) and null.
not		✓	Together with is, contains, endswith, startswith, to negate the condition.
startswith		✓	Starting with a given pattern.

SHQL Filters

How to filter using conditions?

- Conditions are placed within square brackets([ ]). And you can use them to filter results by a specific attribute's value
- Example:

inventory_item [.vendor="Cisco"]



SHQL

Saved Queries | Save

```
inventory_item[.vendor="Cisco"]
```

RUN

RESULTS (1937)

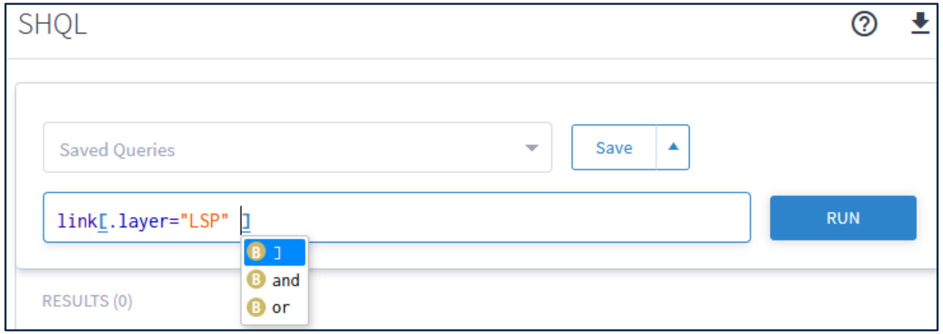
Port (1612) | Power Supply (20) | Fan Tray (50) | Card (176) | Shelf (57) | Optical Node (12) | Router (10)

1612 ITEMS

Guid	Type	Acces	Desc	Mode	Name	Paren	PartN	Provic	Serial	Vendc	Extra	Devic	Equip	Plugg	Suppc	Tunab
IN/o...	PORT	e34d...			POR...	IN/o...		onc		Cisco	{'uui...	IN/o...	INST...	NON...		NON...
IN/o...	PORT	f9d7...			POR...	IN/o...		onc		Cisco	{'uui...	IN/o...	INST...	NON...		NON...
IN/o...	PORT	25f6...			POR...	IN/o...		onc		Cisco	{'uui...	IN/o...	INST...	NON...		NON...
IN/o...	PORT	a917...			POR...	IN/o...		onc		Cisco	{'uui...	IN/o...	INST...	NON...		NON...
IN/o...	PORT	8164...			POR...	IN/o...		onc		Cisco	{'uui...	IN/o...	INST...	NON...		NON...
IN/o...	PORT	f79f9...			POR...	IN/o...		onc		Cisco	{'uui...	IN/o...	INST...	NON...		NON...
IN/o...	PORT	657b...			POR...	IN/o...		onc		Cisco	{'uui...	IN/o...	INST...	NON...		NON...

and\or Operators

- You can combine two or more conditions in order to get specific results, using the logical operator **and/or**
- Example: *link[.layer="LSP" and .name contains "MIL"]*

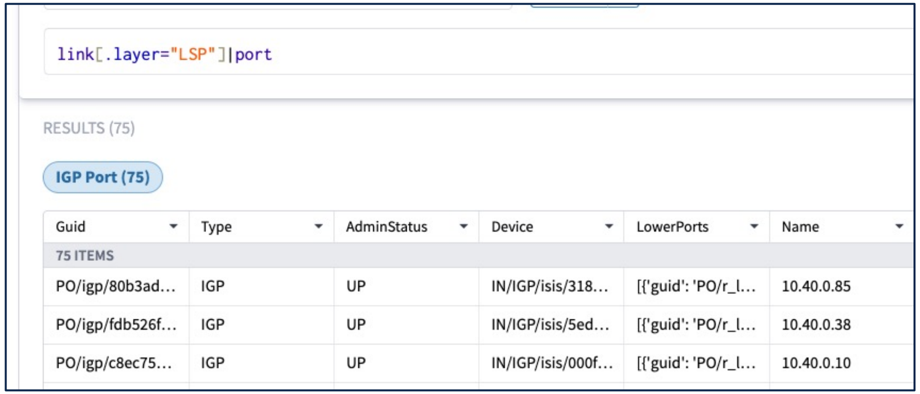


```
link[.layer="LSP" ]
```

SHQL Transformation

Transforming what?

- You can add an object type to the query command and determine if the data to be retrieved is transformed from one object type to another object type, or if the data reflects a collection of multiple object types and their related items.



```
link[.layer="LSP"]|port
```

RESULTS (75)

IGP Port (75)

Guid	Type	AdminStatus	Device	LowerPorts	Name
75 ITEMS					
PO/igp/80b3ad…	IGP	UP	IN/IGP/isis/318…	[{'guid': 'PO/r_l…	10.40.0.85
PO/igp/fdb526f…	IGP	UP	IN/IGP/isis/5ed…	[{'guid': 'PO/r_l…	10.40.0.38
PO/igp/c8ec75…	IGP	UP	IN/IGP/isis/000f…	[{'guid': 'PO/r_l…	10.40.0.10

Transformation Types

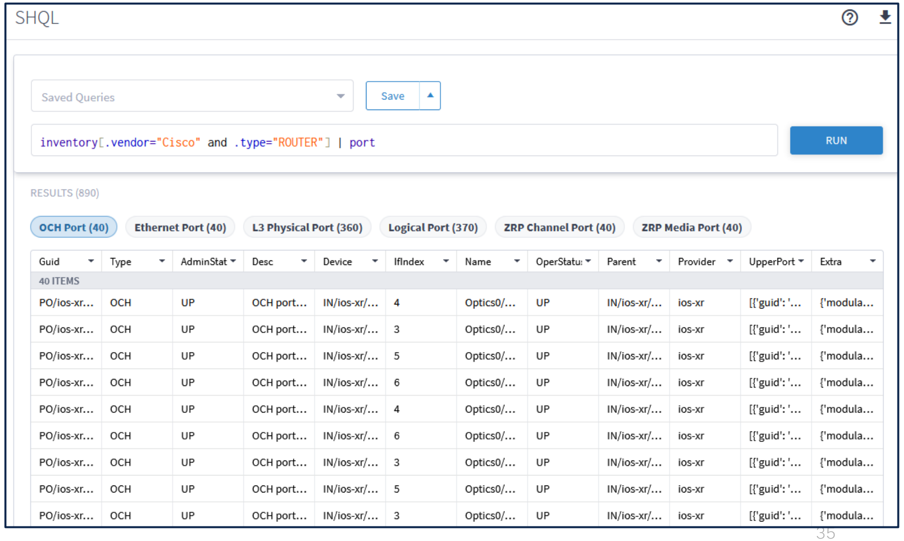
- **Transformation:** Add a pipe (|) to the query command before adding the new object type. Transforms the results relating to the previous object type to output for the new object type.
- **Collection:** Add an ampersand (&) to the query command before adding the new object type. Retrieves all the output for all the preceding object types.
- **As:** Add a temporary variable. Enables you to create a query with an object type that is not related to the preceding object type.

Transformation ‘|’

- Example:

  inventory[.vendor="Cisco" and .type="ROUTER"] | port

- The query will transform the inventory of Cisco routers to the inventory of ports that belong to Cisco routers



```
inventory[.vendor="Cisco" and .type="ROUTER"] | port
```

RESULTS (890)

OCH Port (40) | Ethernet Port (40) | L3 Physical Port (360) | Logical Port (370) | ZRP Channel Port (40) | ZRP Media Port (40)

40 ITEMS

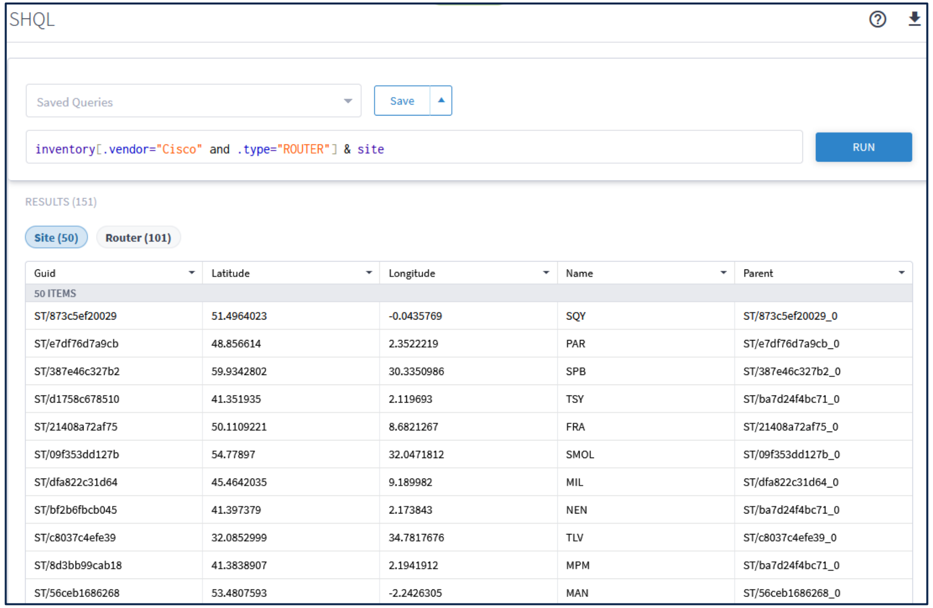
Guid	Type	AdminStat	Desc	Device	IfIndex	Name	OperStatu:	Parent	Provider	UpperPort	Extra
PO/ios-xr...	OCH	UP	OCH port...	IN/ios-xr/...	4	Optics0/...	UP	IN/ios-xr/...	ios-xr	[{'guid': '...	{'modula...
PO/ios-xr...	OCH	UP	OCH port...	IN/ios-xr/...	3	Optics0/...	UP	IN/ios-xr/...	ios-xr	[{'guid': '...	{'modula...
PO/ios-xr...	OCH	UP	OCH port...	IN/ios-xr/...	5	Optics0/...	UP	IN/ios-xr/...	ios-xr	[{'guid': '...	{'modula...
PO/ios-xr...	OCH	UP	OCH port...	IN/ios-xr/...	6	Optics0/...	UP	IN/ios-xr/...	ios-xr	[{'guid': '...	{'modula...
PO/ios-xr...	OCH	UP	OCH port...	IN/ios-xr/...	4	Optics0/...	UP	IN/ios-xr/...	ios-xr	[{'guid': '...	{'modula...
PO/ios-xr...	OCH	UP	OCH port...	IN/ios-xr/...	6	Optics0/...	UP	IN/ios-xr/...	ios-xr	[{'guid': '...	{'modula...
PO/ios-xr...	OCH	UP	OCH port...	IN/ios-xr/...	3	Optics0/...	UP	IN/ios-xr/...	ios-xr	[{'guid': '...	{'modula...
PO/ios-xr...	OCH	UP	OCH port...	IN/ios-xr/...	5	Optics0/...	UP	IN/ios-xr/...	ios-xr	[{'guid': '...	{'modula...
PO/ios-xr...	OCH	UP	OCH port...	IN/ios-xr/...	3	Optics0/...	UP	IN/ios-xr/...	ios-xr	[{'guid': '...	{'modula...

Collection ‘&’

- The next query is an example:

  inventory[.vendor="Cisco" and .type="ROUTER"] & site

- The query will get the Cisco routers **and** the sites where those Cisco routers belong.



SHQL

Saved Queries | Save | RUN

```
inventory[.vendor="Cisco" and .type="ROUTER"] & site
```

RESULTS (151)

Site (50) | Router (101)

50 ITEMS

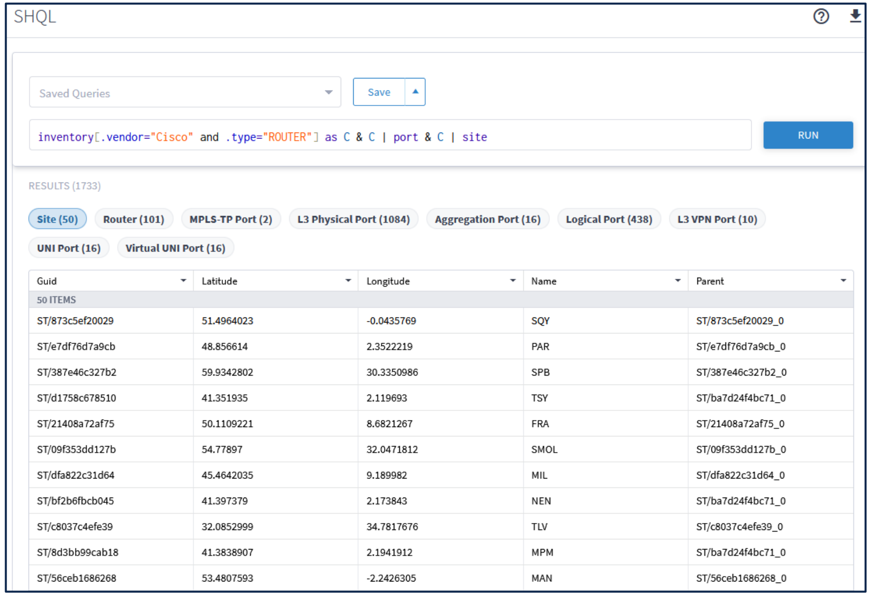
Guid	Latitude	Longitude	Name	Parent
ST/873c5ef20029	51.4964023	-0.0435769	SQY	ST/873c5ef20029_0
ST/e7df76d7a9cb	48.856614	2.3522219	PAR	ST/e7df76d7a9cb_0
ST/387e46c327b2	59.9342802	30.3350986	SPB	ST/387e46c327b2_0
ST/d1758c678510	41.351935	2.119693	TSY	ST/ba7d24f4bc71_0
ST/21408a72af75	50.1109221	8.6821267	FRA	ST/21408a72af75_0
ST/09f353dd127b	54.77897	32.0471812	SMOL	ST/09f353dd127b_0
ST/dfa822c31d64	45.4642035	9.189982	MIL	ST/dfa822c31d64_0
ST/bf2b6fbcb045	41.397379	2.173843	NEN	ST/ba7d24f4bc71_0
ST/c8037c4efe39	32.0852999	34.7817676	TLV	ST/c8037c4efe39_0
ST/8d3bb99cab18	41.3838907	2.1941912	MPM	ST/ba7d24f4bc71_0
ST/56ceb1686268	53.4807593	-2.2426305	MAN	ST/56ceb1686268_0

Temporary Variable

- The next query is an example:

  inventory[.vendor="Cisco" and .type="ROUTER"] as C & C | port & C | site

- The query will transform the inventory of Cisco routers to the inventory of ports that belong to Cisco routers and sites where Cisco routers exists.



```
inventory[.vendor="Cisco" and .type="ROUTER"] as C & C | port & C | site
```

RESULTS (1733)

Site (50) | Router (101) | MPLS-TP Port (2) | L3 Physical Port (1084) | Aggregation Port (16) | Logical Port (438) | L3 VPN Port (10) | UNI Port (16) | Virtual UNI Port (16)

50 ITEMS

Guid	Latitude	Longitude	Name	Parent
ST/873c5ef20029	51.4964023	-0.0435769	SQY	ST/873c5ef20029_0
ST/e7df76d7a9cb	48.856614	2.3522219	PAR	ST/e7df76d7a9cb_0
ST/387e46c327b2	59.9342802	30.3350986	SPB	ST/387e46c327b2_0
ST/d1758c678510	41.351935	2.119693	TSY	ST/ba7d24f4bc71_0
ST/21408a72af75	50.1109221	8.6821267	FRA	ST/21408a72af75_0
ST/09f353dd127b	54.77897	32.0471812	SMOL	ST/09f353dd127b_0
ST/dfa822c31d64	45.4642035	9.189982	MIL	ST/dfa822c31d64_0
ST/bf2b6fbcb045	41.397379	2.173843	NEN	ST/ba7d24f4bc71_0
ST/c8037c4efe39	32.0852999	34.7817676	TLV	ST/c8037c4efe39_0
ST/8d3bb99cab18	41.3838907	2.1941912	MPM	ST/ba7d24f4bc71_0
ST/56ceb1686268	53.4807593	-2.2426305	MAN	ST/56ceb1686268_0

SHQL Function

What is it for?

- Functions are preceded by a pipe in the query command line.
- You can retrieve an item and then specify whether to retrieve related items from either above or below the layer, or from both above and below. These recursive operations are valid for port, link, site, inventory, and visual site.

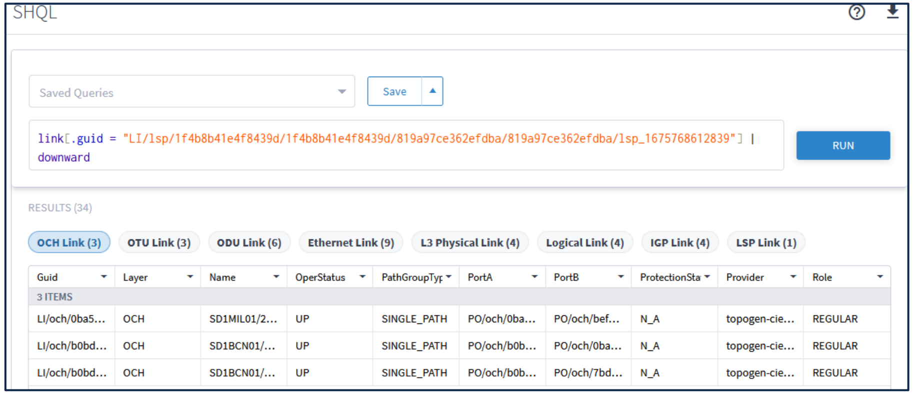
Function	Description
Downward	Retrieves items from below the layer of the specified item
Upward	Retrieves items from above the layer of the specified item
Span	Retrieves items from below and above the layer of the specified item
FTS	Free text search. Retrieves items according to the search string you enter.
Retrospective(@)	Retrieves items from the past according to a given timestamp.

downward

- The next query is an example for downward function:

  link[.guid=" LI/lsp/1f4b8b41e4f8439d/1f4b8b41e4f8439d/819a97ce362efdba/819a97ce362efdba/lsp_1675768612839"] | downward

- The query will retrieve information from all layers bellow the selected LSP link.



```
link[.guid = "LI/lsp/1f4b8b41e4f8439d/1f4b8b41e4f8439d/819a97ce362efdba/819a97ce362efdba/lsp_1675768612839"] |
downward
```

RESULTS (34)

OCH Link (3) | OTU Link (3) | ODU Link (6) | Ethernet Link (9) | L3 Physical Link (4) | Logical Link (4) | IGP Link (4) | LSP Link (1)

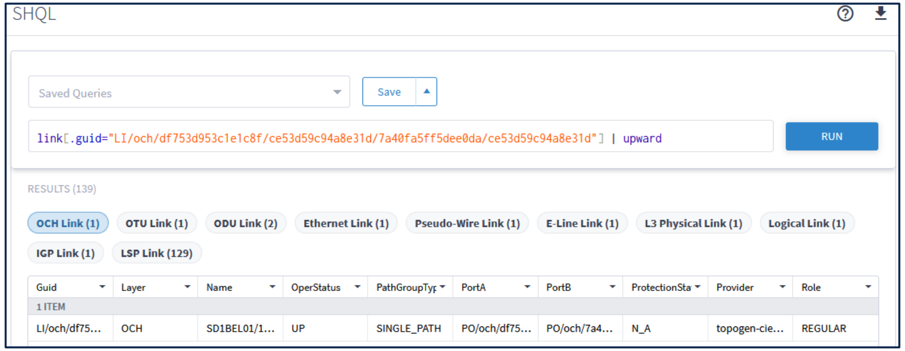
Guid	Layer	Name	OperStatus	PathGroupTyp	PortA	PortB	ProtectionSta	Provider	Role
3 ITEMS									
LI/och/0ba5...	OCH	SD1MIL01/2...	UP	SINGLE_PATH	PO/och/0ba...	PO/och/bef...	N_A	topogen-cie...	REGULAR
LI/och/b0bd...	OCH	SD1BCN01/...	UP	SINGLE_PATH	PO/och/b0b...	PO/och/0ba...	N_A	topogen-cie...	REGULAR
LI/och/b0bd...	OCH	SD1BCN01/...	UP	SINGLE_PATH	PO/och/b0b...	PO/och/7bd...	N_A	topogen-cie...	REGULAR

upward

- The next query is an example for upward function:

  link[.guid="LI/och/df753d953c1e1c8f/ce53d59c94a8e31d/7a40fa5ff5dee0da/ce53d59c94a8e31d"] | upward

- The query will retrieve information from all layers above the selected item.



```
link[.guid="LI/och/df753d953c1e1c8f/ce53d59c94a8e31d/7a40fa5ff5dee0da/ce53d59c94a8e31d"] | upward
```

RESULTS (139)

OCH Link (1) · OTU Link (1) · ODU Link (2) · Ethernet Link (1) · Pseudo-Wire Link (1) · E-Line Link (1) · L3 Physical Link (1) · Logical Link (1) · IGP Link (1) · LSP Link (129)

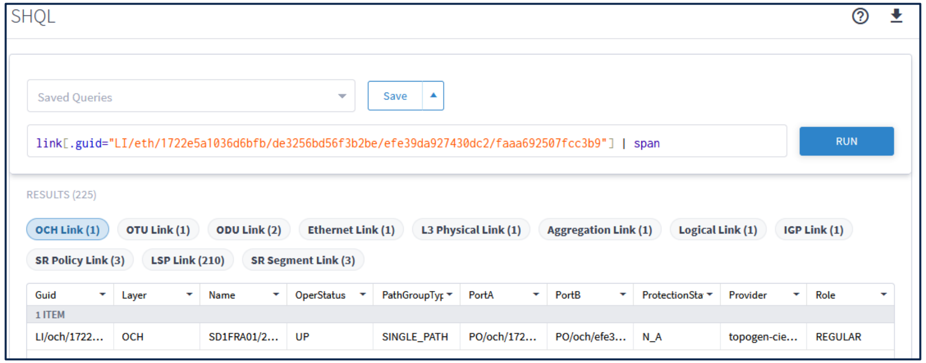
Guid	Layer	Name	OperStatus	PathGroupTyp	PortA	PortB	ProtectionSta	Provider	Role
1 ITEM									
LI/och/df75...	OCH	SD1BEL01/1...	UP	SINGLE_PATH	PO/och/df75...	PO/och/7a4...	N_A	topogen-cie...	REGULAR

span

- The next query is an example for span function:

  link[.guid="LI/eth/1722e5a1036d6bfb/de3256bd56f3b2be/efe39da927430dc2/faaa692507fcc3b9"] | span

- The query will retrieve information from all layer bellow and above the selected item.


SHQL

```
link[.guid="LI/eth/1722e5a1036d6bfb/de3256bd56f3b2be/efe39da927430dc2/faaa692507fcc3b9"] | span
```

RESULTS (225)

OCH Link (1) OTU Link (1) ODU Link (2) Ethernet Link (1) L3 Physical Link (1) Aggregation Link (1) Logical Link (1) IGP Link (1) SR Policy Link (3) LSP Link (210) SR Segment Link (3)

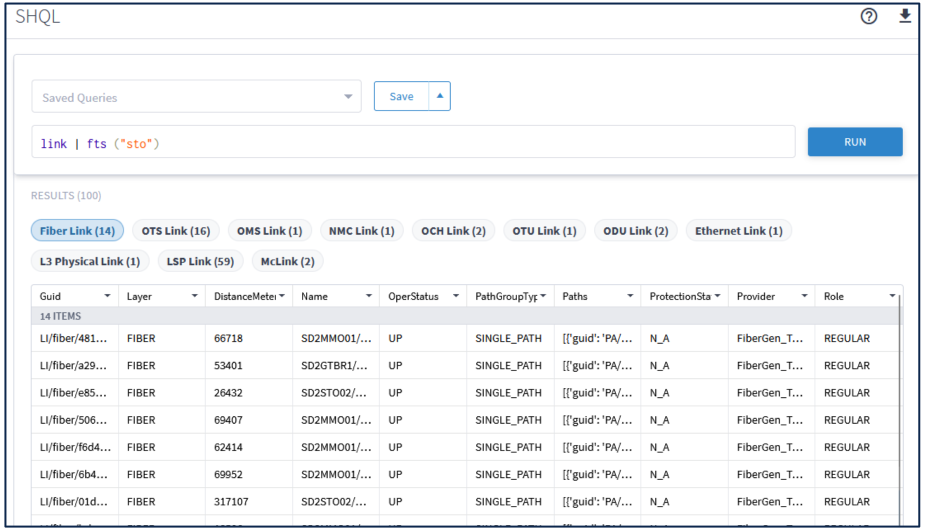
Guid	Layer	Name	OperStatus	PathGroupTyp	PortA	PortB	ProtectionSta	Provider	Role
1 ITEM									
LI/och/1722...	OCH	SD1FRA01/2...	UP	SINGLE_PATH	PO/och/172...	PO/och/efe3...	N_A	topogen-cie...	REGULAR

Free Text

- The next query is an example for free text search function:

  link | fts (”sto")

- The query will retrieve information of all links that contains the string “sto” in any of its fields.
- It is not a recommended way as it may bring a lot of irrelevant objects in the response



SHQL

Saved Queries

Save

```
link | fts ("sto")
```

RUN

RESULTS (100)

Fiber Link (14) | OTS Link (16) | OMS Link (1) | NMC Link (1) | OCH Link (2) | OTU Link (1) | ODU Link (2) | Ethernet Link (1) | L3 Physical Link (1) | LSP Link (59) | McLink (2)

Guid	Layer	DistanceMeter	Name	OperStatus	PathGroupTyp	Paths	ProtectionSta	Provider	Role
14 ITEMS									
LI/fiber/481...	FIBER	66718	SD2MMO01/...	UP	SINGLE_PATH	[{'guid': 'PA/...	N_A	FiberGen_T...	REGULAR
LI/fiber/a29...	FIBER	53401	SD2GTBR1/...	UP	SINGLE_PATH	[{'guid': 'PA/...	N_A	FiberGen_T...	REGULAR
LI/fiber/e85...	FIBER	26432	SD2STO02/...	UP	SINGLE_PATH	[{'guid': 'PA/...	N_A	FiberGen_T...	REGULAR
LI/fiber/506...	FIBER	69407	SD2MMO01/...	UP	SINGLE_PATH	[{'guid': 'PA/...	N_A	FiberGen_T...	REGULAR
LI/fiber/f6d4...	FIBER	62414	SD2MMO01/...	UP	SINGLE_PATH	[{'guid': 'PA/...	N_A	FiberGen_T...	REGULAR
LI/fiber/6b4...	FIBER	69952	SD2MMO01/...	UP	SINGLE_PATH	[{'guid': 'PA/...	N_A	FiberGen_T...	REGULAR
LI/fiber/01d...	FIBER	317107	SD2STO02/...	UP	SINGLE_PATH	[{'guid': 'PA/...	N_A	FiberGen_T...	REGULAR

Time Based Queries

Time Machine (Retrospective)

- Absolute time: @2019-05-10 10:00:00
- Relative time in the format:

  '-'[0-9]+[ymwdHMS]: @-10H
- Unix timestamp (ms): @1558610956000
- The next query is an example for retrospective function:

  @-40d link[.operStatus="DOWN"]
- The query will retrieve information of all links that were down at specified time, in this example 40 days ago.

Again, what?

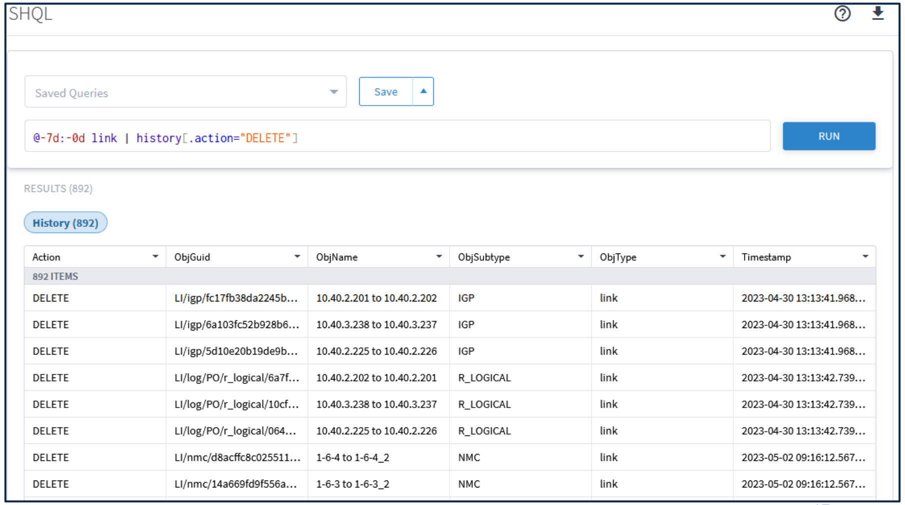
- HCO keeps records of all changes in the network inventory and topology. The changes are stored in the Database. Every change will be stored as a new record with the timestamp. A change is a record of any resource addition (ADD), deletion (DELETE) or attribute change (UPDATE).
- You can construct a query that uses a standard SHQL query to filter the model, then add the pipe (|) and filter the history table.

History Data

Example:

@-7d:-0d link | history[.action="UPDATE"]

The query will retrieve all the items that have a change during specified time span, for this example 7 days in the past.


SHQL

Saved Queries

Save

```
@-7d:-0d link | history[.action="DELETE"]
```

RUN

RESULTS (892)

History (892)

892 ITEMS

Action	ObjGuid	ObjName	ObjSubtype	ObjType	Timestamp
DELETE	LI/igp/fc17fb38da2245b…	10.40.2.201 to 10.40.2.202	IGP	link	2023-04-30 13:13:41.968…
DELETE	LI/igp/6a103fc52b928b6…	10.40.3.238 to 10.40.3.237	IGP	link	2023-04-30 13:13:41.968…
DELETE	LI/igp/5d10e20b19de9b…	10.40.2.225 to 10.40.2.226	IGP	link	2023-04-30 13:13:41.968…
DELETE	LI/log/PO/r_logical/6a7f…	10.40.2.202 to 10.40.2.201	R_LOGICAL	link	2023-04-30 13:13:42.739…
DELETE	LI/log/PO/r_logical/10cf…	10.40.3.238 to 10.40.3.237	R_LOGICAL	link	2023-04-30 13:13:42.739…
DELETE	LI/log/PO/r_logical/064…	10.40.2.225 to 10.40.2.226	R_LOGICAL	link	2023-04-30 13:13:42.739…
DELETE	LI/nmc/d8acffc8c025511…	1-6-4 to 1-6-4_2	NMC	link	2023-05-02 09:16:12.567…
DELETE	LI/nmc/14a669fd9f556a…	1-6-3 to 1-6-3_2	NMC	link	2023-05-02 09:16:12.567…

SHQL Output Functions

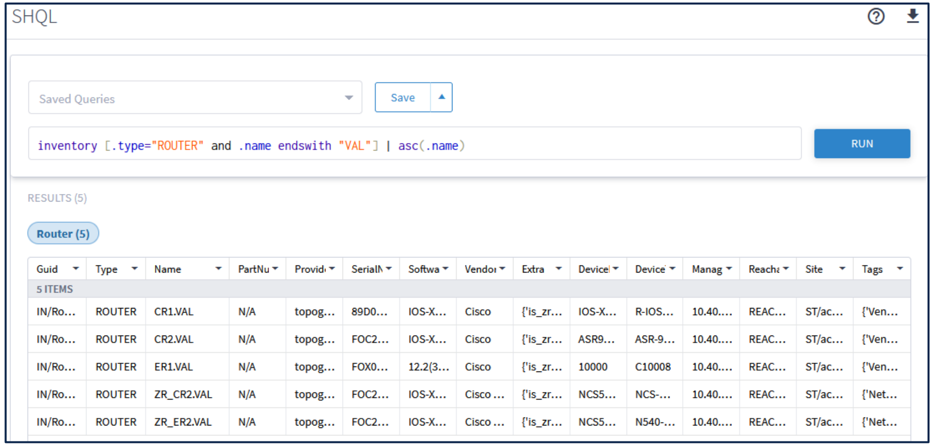
SHQL Output Functions

- You can add the functions to retrieve results and display them in a specific order. Typically, these functions are added at the end of the query command. You can also view specific properties for the query results.

Function	**Description**
asc (column)	Displays results in ascending natural order
desc (column)	Displays results in descending natural order
limit(#)	Limits the number of displayed results
after (GUID)	Displays only the results that follow the item with the specified GUID.
add_counters	Displays the total number per attribute value for the specified object type.
view	Displays the specified properties (with the labels provided) for the query results.

Ascending

- Example:
  inventory [.type="ROUTER" and .name endswith "VAL"] | asc(.name)
- The query will retrieve information of all Router which names end in “VAL” and order the results in ascending natural order.



SHQL

Saved Queries | Save | RUN

```
inventory [.type="ROUTER" and .name endswith "VAL"] | asc(.name)
```

RESULTS (5)

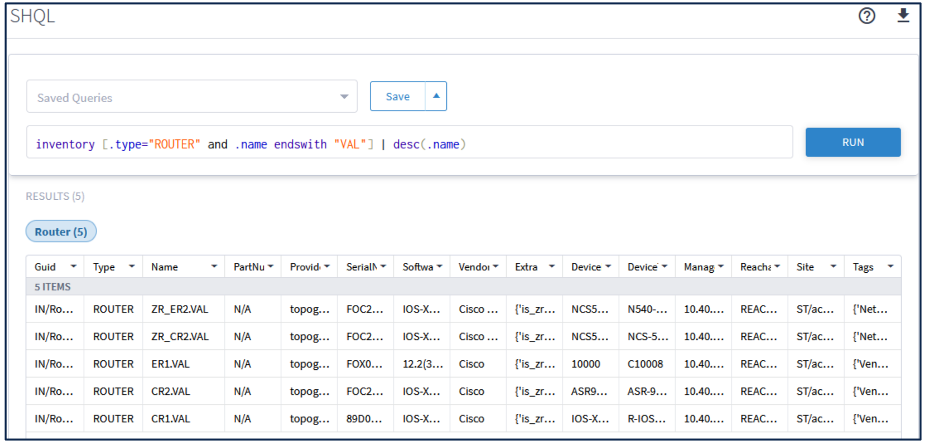
Router (5)

5 ITEMS

Guid	Type	Name	PartNu	Provid	SerialN	Softwa	Vendor	Extra	Device	Device	Manag	Reacha	Site	Tags
IN/Ro...	ROUTER	CR1.VAL	N/A	topog...	89D0...	IOS-X...	Cisco	{'is_zr...	IOS-X...	R-IOS...	10.40....	REAC...	ST/ac...	{'Ven...
IN/Ro...	ROUTER	CR2.VAL	N/A	topog...	FOC2...	IOS-X...	Cisco	{'is_zr...	ASR9...	ASR-9...	10.40....	REAC...	ST/ac...	{'Ven...
IN/Ro...	ROUTER	ER1.VAL	N/A	topog...	FOX0...	12.2(3...	Cisco	{'is_zr...	10000	C10008	10.40....	REAC...	ST/ac...	{'Ven...
IN/Ro...	ROUTER	ZR_CR2.VAL	N/A	topog...	FOC2...	IOS-X...	Cisco ...	{'is_zr...	NCS5...	NCS-...	10.40....	REAC...	ST/ac...	{'Net...
IN/Ro...	ROUTER	ZR_ER2.VAL	N/A	topog...	FOC2...	IOS-X...	Cisco ...	{'is_zr...	NCS5...	N540-...	10.40....	REAC...	ST/ac...	{'Net...

Descending

- Example:
  inventory [.type="ROUTER" and .name endswith "VAL"] | desc(.name)
- The query will retrieve information of all Router which names end in “VAL” and order the results in descending natural order.



SHQL

Saved Queries

```
inventory [.type="ROUTER" and .name endswith "VAL"] | desc(.name)
```

RESULTS (5)

Router (5)

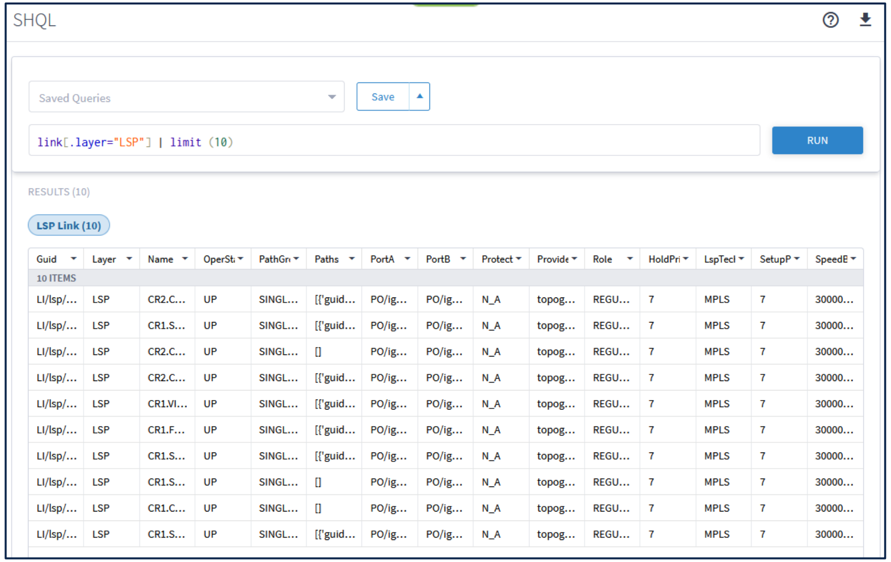
Guid	Type	Name	PartNu	Provid	SerialN	Softwa	Vendor	Extra	Device	Device	Manag	Reacha	Site	Tags
5 ITEMS														
IN/Ro...	ROUTER	ZR_ER2.VAL	N/A	topog...	FOC2...	IOS-X...	Cisco ...	{'is_zr...	NCS5...	N540-...	10.40....	REAC...	ST/ac...	{'Net...
IN/Ro...	ROUTER	ZR_CR2.VAL	N/A	topog...	FOC2...	IOS-X...	Cisco ...	{'is_zr...	NCS5...	NCS-5...	10.40....	REAC...	ST/ac...	{'Net...
IN/Ro...	ROUTER	ER1.VAL	N/A	topog...	FOX0...	12.2(3...	Cisco	{'is_zr...	10000	C10008	10.40....	REAC...	ST/ac...	{'Ven...
IN/Ro...	ROUTER	CR2.VAL	N/A	topog...	FOC2...	IOS-X...	Cisco	{'is_zr...	ASR9...	ASR-9...	10.40....	REAC...	ST/ac...	{'Ven...
IN/Ro...	ROUTER	CR1.VAL	N/A	topog...	89D0...	IOS-X...	Cisco	{'is_zr...	IOS-X...	R-IOS...	10.40....	REAC...	ST/ac...	{'Ven...

Limit

- Example:

  link[.layer="LSP"] | limit (10)

- The query will retrieve information of all LSP links and limit to display to first 10 items.


SHQL

Saved Queries | Save | RUN

```
link[.layer="LSP"] | limit (10)
```

RESULTS (10)

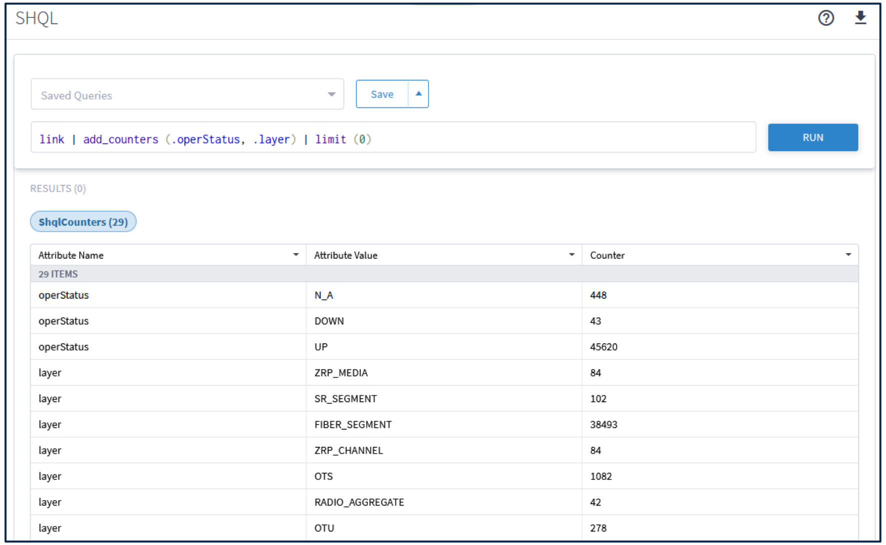
LSP Link (10)

10 ITEMS

Guid	Layer	Name	OperSt	PathGr	Paths	PortA	PortB	Protect	Provide	Role	HoldPri	LspTecl	SetupP	SpeedE
LI/lsp/...	LSP	CR2.C...	UP	SINGL...	[{'guid...	PO/ig...	PO/ig...	N_A	topog...	REGU...	7	MPLS	7	30000...
LI/lsp/...	LSP	CR1.S...	UP	SINGL...	[{'guid...	PO/ig...	PO/ig...	N_A	topog...	REGU...	7	MPLS	7	30000...
LI/lsp/...	LSP	CR2.C...	UP	SINGL...	[]	PO/ig...	PO/ig...	N_A	topog...	REGU...	7	MPLS	7	30000...
LI/lsp/...	LSP	CR2.C...	UP	SINGL...	[{'guid...	PO/ig...	PO/ig...	N_A	topog...	REGU...	7	MPLS	7	30000...
LI/lsp/...	LSP	CR1.VI...	UP	SINGL...	[{'guid...	PO/ig...	PO/ig...	N_A	topog...	REGU...	7	MPLS	7	30000...
LI/lsp/...	LSP	CR1.F...	UP	SINGL...	[{'guid...	PO/ig...	PO/ig...	N_A	topog...	REGU...	7	MPLS	7	30000...
LI/lsp/...	LSP	CR1.S...	UP	SINGL...	[{'guid...	PO/ig...	PO/ig...	N_A	topog...	REGU...	7	MPLS	7	30000...
LI/lsp/...	LSP	CR1.S...	UP	SINGL...	[]	PO/ig...	PO/ig...	N_A	topog...	REGU...	7	MPLS	7	30000...
LI/lsp/...	LSP	CR1.C...	UP	SINGL...	[]	PO/ig...	PO/ig...	N_A	topog...	REGU...	7	MPLS	7	30000...
LI/lsp/...	LSP	CR1.S...	UP	SINGL...	[{'guid...	PO/ig...	PO/ig...	N_A	topog...	REGU...	7	MPLS	7	30000...

Objects' Counter

- Example:
  link | add_counters (.operStatus, .layer) | limit (0)
- The query will retrieve the counts of Operational Status and layers of links.



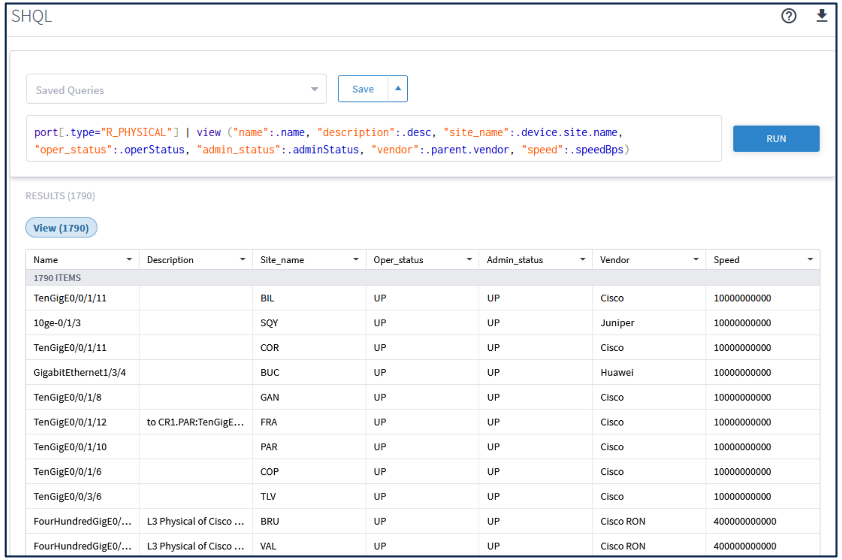
```
link | add_counters (.operStatus, .layer) | limit (0)
```

29 ITEMS

Attribute Name	Attribute Value	Counter
operStatus	N_A	448
operStatus	DOWN	43
operStatus	UP	45620
layer	ZRP_MEDIA	84
layer	SR_SEGMENT	102
layer	FIBER_SEGMENT	38493
layer	ZRP_CHANNEL	84
layer	OTS	1082
layer	RADIO_AGGREGATE	42
layer	OTU	278

view

- Example:
  port[.type="R_PHYSICAL"] | view ("name":.name, "description":.desc, "site_name":.device.site.name, "oper_status":.operStatus, "admin_status":.adminStatus, "vendor":.parent.vendor, "speed":.speedBps)
- The query will retrieve the information with the order and information input in query.



```
port[.type="R_PHYSICAL"] | view ("name":.name, "description":.desc, "site_name":.device.site.name,
"oper_status":.operStatus, "admin_status":.adminStatus, "vendor":.parent.vendor, "speed":.speedBps)
```

RESULTS (1790)

View (1790)

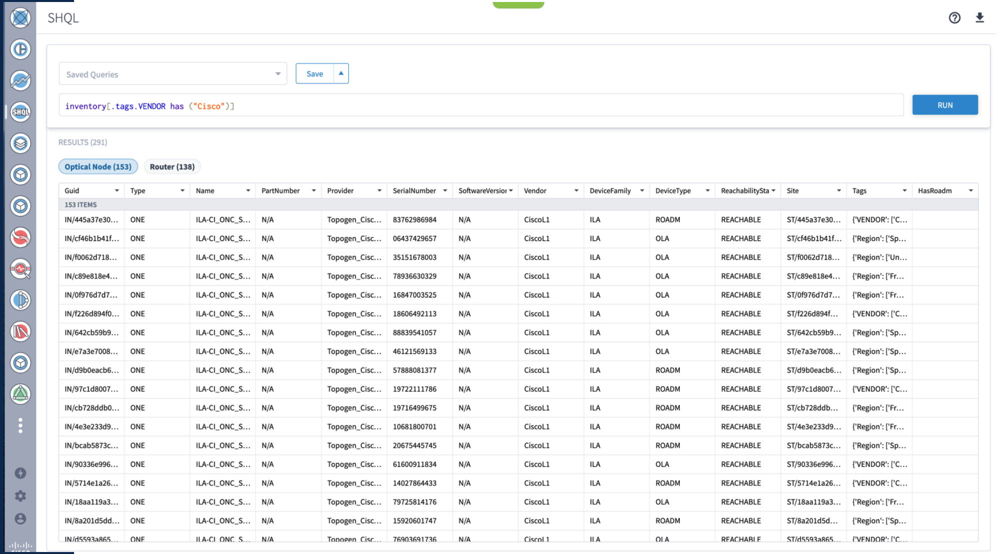
Name	Description	Site_name	Oper_status	Admin_status	Vendor	Speed
1790 ITEMS						
TenGigE0/0/1/11		BIL	UP	UP	Cisco	10000000000
10ge-0/1/3		SQY	UP	UP	Juniper	10000000000
TenGigE0/0/1/11		COR	UP	UP	Cisco	10000000000
GigabitEthernet1/3/4		BUC	UP	UP	Huawei	10000000000
TenGigE0/0/1/8		GAN	UP	UP	Cisco	10000000000
TenGigE0/0/1/12	to CR1.PAR:TenGigE…	FRA	UP	UP	Cisco	10000000000
TenGigE0/0/1/10		PAR	UP	UP	Cisco	10000000000
TenGigE0/0/1/6		COP	UP	UP	Cisco	10000000000
TenGigE0/0/3/6		TLV	UP	UP	Cisco	10000000000
FourHundredGigE0/…	L3 Physical of Cisco …	BRU	UP	UP	Cisco RON	400000000000
FourHundredGigE0/…	L3 Physical of Cisco …	VAL	UP	UP	Cisco RON	400000000000

Tags and Regions

Tags

- An example for using tags to query devices assigned to a specific tag, in this example we will get all the devices assigned with a tag key VENDOR and Value Cisco:

```
inventory[.tags.VENDOR has ("Cisco")]
```



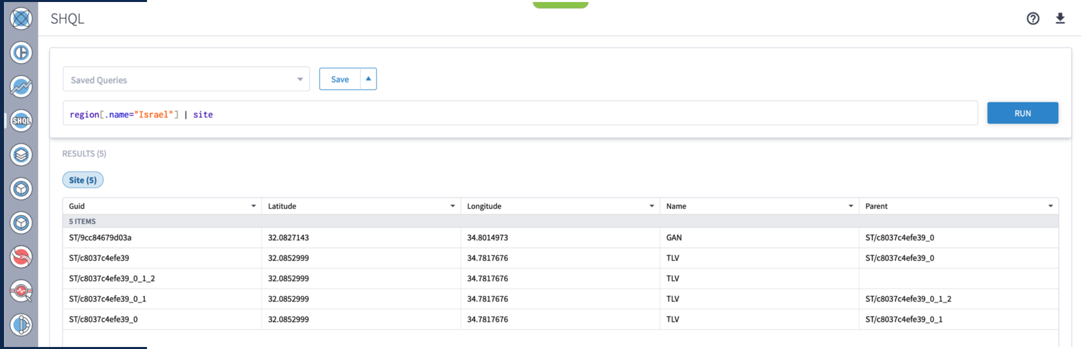
```
inventory[.tags.VENDOR has ("Cisco")]
```

RESULTS (291)

Guid	Type	Name	PartNumber	Provider	SerialNumber	SoftwareVersioı	Vendor	DeviceFamily	DeviceType	ReachabilitySta	Site	Tags	HasRoadm
153 ITEMS													
IN/445a37e30...	ONE	ILA-CI_ONC_S...	N/A	Topogen_Cisc...	83762986984	N/A	CiscoL1	ILA	ROADM	REACHABLE	ST/445a37e30...	{'VENDOR': ['C...	
IN/cf46b1b41f...	ONE	ILA-CI_ONC_S...	N/A	Topogen_Cisc...	06437429657	N/A	CiscoL1	ILA	OLA	REACHABLE	ST/cf46b1b41f...	{'Region': ['Sp...	
IN/f0062d718...	ONE	ILA-CI_ONC_S...	N/A	Topogen_Cisc...	35151678003	N/A	CiscoL1	ILA	OLA	REACHABLE	ST/f0062d718...	{'Region': ['Un...	
IN/c89e818e4...	ONE	ILA-CI_ONC_S...	N/A	Topogen_Cisc...	78936630329	N/A	CiscoL1	ILA	OLA	REACHABLE	ST/c89e818e4...	{'Region': ['Fr...	
IN/0f976d7d7...	ONE	ILA-CI_ONC_S...	N/A	Topogen_Cisc...	16847003525	N/A	CiscoL1	ILA	OLA	REACHABLE	ST/0f976d7d7...	{'Region': ['Fr...	
IN/f226d894f0...	ONE	ILA-CI_ONC_S...	N/A	Topogen_Cisc...	18606492113	N/A	CiscoL1	ILA	OLA	REACHABLE	ST/f226d894f...	{'VENDOR': ['C...	
IN/642cb59b9...	ONE	ILA-CI_ONC_S...	N/A	Topogen_Cisc...	88839541057	N/A	CiscoL1	ILA	OLA	REACHABLE	ST/642cb59b9...	{'Region': ['Sp...	
IN/e7a3e7008...	ONE	ILA-CI_ONC_S...	N/A	Topogen_Cisc...	46121569133	N/A	CiscoL1	ILA	OLA	REACHABLE	ST/e7a3e7008...	{'Region': ['Sp...	
IN/d9b0eacb6...	ONE	ILA-CI_ONC_S...	N/A	Topogen_Cisc...	57888081377	N/A	CiscoL1	ILA	ROADM	REACHABLE	ST/d9b0eacb6...	{'Region': ['Sp...	
IN/97c1d8007...	ONE	ILA-CI_ONC_S...	N/A	Topogen_Cisc...	19722111786	N/A	CiscoL1	ILA	ROADM	REACHABLE	ST/97c1d8007...	{'VENDOR': ['C...	
IN/cb728ddb0...	ONE	ILA-CI_ONC_S...	N/A	Topogen_Cisc...	19716499675	N/A	CiscoL1	ILA	ROADM	REACHABLE	ST/cb728ddb...	{'Region': ['Fr...	
IN/4e3e233d9...	ONE	ILA-CI_ONC_S...	N/A	Topogen_Cisc...	10681800701	N/A	CiscoL1	ILA	ROADM	REACHABLE	ST/4e3e233d9...	{'Region': ['Fr...	
IN/bcab5873c...	ONE	ILA-CI_ONC_S...	N/A	Topogen_Cisc...	20675445745	N/A	CiscoL1	ILA	ROADM	REACHABLE	ST/bcab5873c...	{'Region': ['Sp...	
IN/90336e996...	ONE	ILA-CI_ONC_S...	N/A	Topogen_Cisc...	61600911834	N/A	CiscoL1	ILA	OLA	REACHABLE	ST/90336e996...	{'VENDOR': ['C...	
IN/5714e1a26...	ONE	ILA-CI_ONC_S...	N/A	Topogen_Cisc...	14027864433	N/A	CiscoL1	ILA	ROADM	REACHABLE	ST/5714e1a26...	{'VENDOR': ['C...	
IN/18aa119a3...	ONE	ILA-CI_ONC_S...	N/A	Topogen_Cisc...	79725814176	N/A	CiscoL1	ILA	OLA	REACHABLE	ST/18aa119a3...	{'Region': ['Fr...	
IN/8a201d5dd...	ONE	ILA-CI_ONC_S...	N/A	Topogen_Cisc...	15920601747	N/A	CiscoL1	ILA	ROADM	REACHABLE	ST/8a201d5d...	{'Region': ['Sp...	
IN/d5593a865...	ONE	ILA-CI_ONC_S...	N/A	Topogen_Cisc...	76903691736	N/A	CiscoL1	ILA	OLA	REACHABLE	ST/d5593a865...	{'VENDOR': ['C...	

Regions

- An example for using regions to query sites assigned to a specific region or overlay, in this example we will get all the sites assigned to region Israel:
- region[.name="Israel"] | site



```
region[.name="Israel"] | site
```

5 ITEMS

Guid	Latitude	Longitude	Name	Parent
ST/9cc84679d03a	32.0827143	34.8014973	GAN	ST/c8037c4efe39_0
ST/c8037c4efe39	32.0852999	34.7817676	TLV	ST/c8037c4efe39_0
ST/c8037c4efe39_0_1_2	32.0852999	34.7817676	TLV	
ST/c8037c4efe39_0_1	32.0852999	34.7817676	TLV	ST/c8037c4efe39_0_1_2
ST/c8037c4efe39_0	32.0852999	34.7817676	TLV	ST/c8037c4efe39_0_1

Tags and Regions

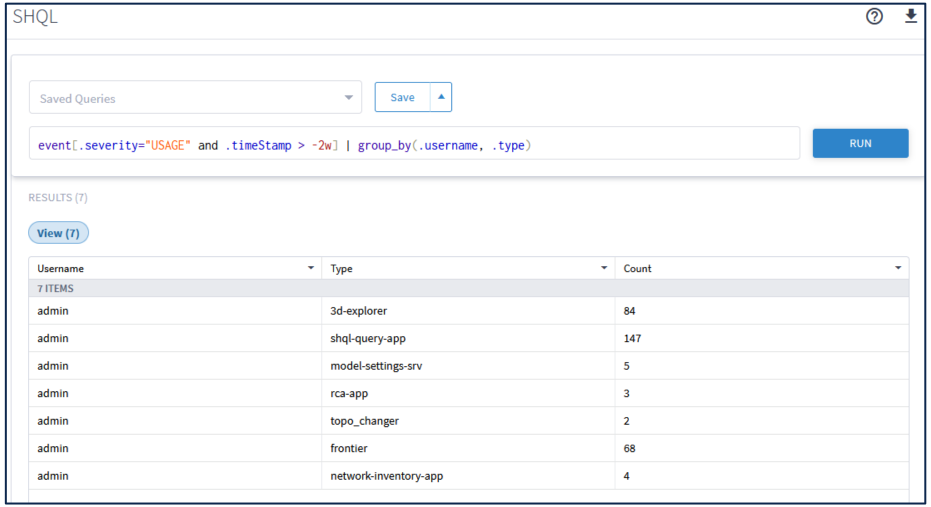
- An example for using tags and regions to query Network resources assigned to a specific tag and region, in this example we will get all the physical ethernet ports from devices assigned with a tag key VENDOR and Value Cisco and belonging to region Israel.
- region[.name="Israel"] | site | inventory[.tags.VENDOR has ("Cisco")] | port[.type="R_PHYSICAL"]

Events

Events Queries

- events can be for usage by users and applications, information, debug, etc.
- Example for using events, in this case we will search events of user activity.

event[.severity="USAGE" and .timeStamp > -2w] | group_by(.username, .type)



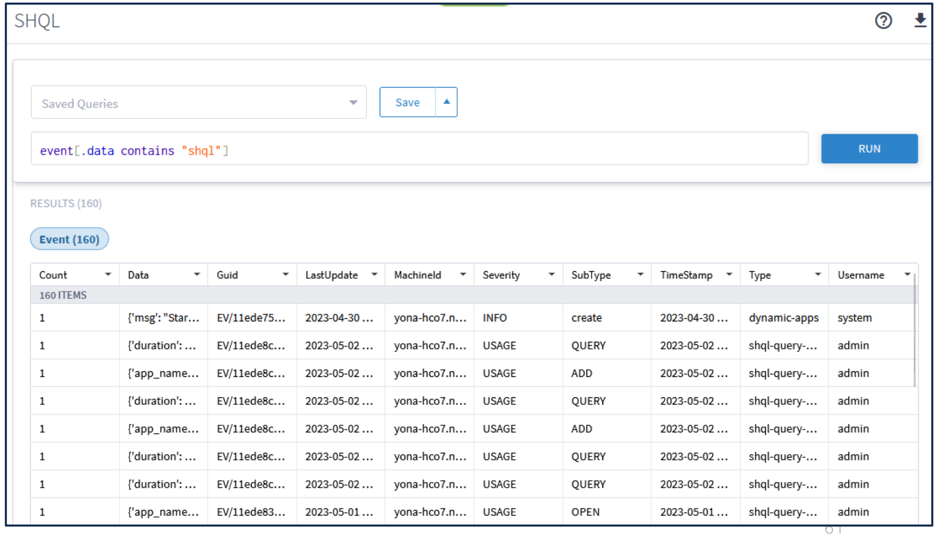
```
event[.severity="USAGE" and .timeStamp > -2w] | group_by(.username, .type)
```

7 ITEMS

Username	Type	Count
admin	3d-explorer	84
admin	shql-query-app	147
admin	model-settings-srv	5
admin	rca-app	3
admin	topo_changer	2
admin	frontier	68
admin	network-inventory-app	4

Another Example

- The next query is an example for using events, in this case we will search events that contains the string “shql” in data field.
- event[.data contains "shql"]



```
event[.data contains "shql"]
```

RESULTS (160)

Event (160)

160 ITEMS

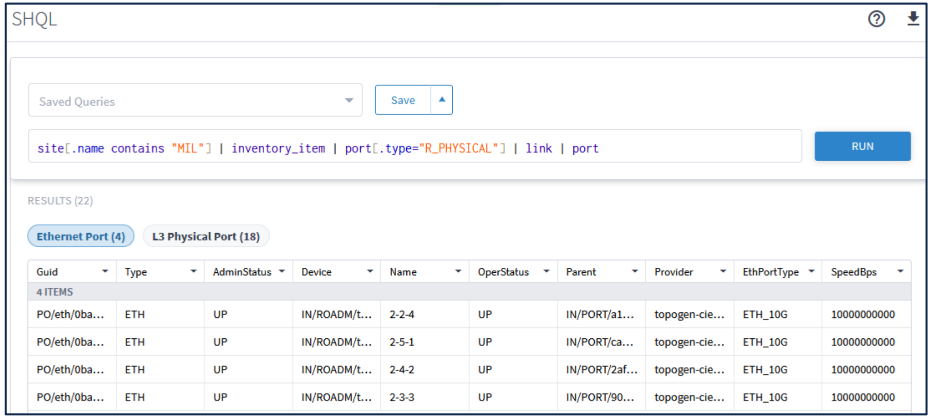
Count	Data	Guid	LastUpdate	MachineId	Severity	SubType	TimeStamp	Type	Username
1	{'msg': "Star…	EV/11ede75…	2023-04-30 …	yona-hco7.n…	INFO	create	2023-04-30 …	dynamic-apps	system
1	{'duration': …	EV/11ede8c…	2023-05-02 …	yona-hco7.n…	USAGE	QUERY	2023-05-02 …	shql-query-…	admin
1	{'app_name…	EV/11ede8c…	2023-05-02 …	yona-hco7.n…	USAGE	ADD	2023-05-02 …	shql-query-…	admin
1	{'duration': …	EV/11ede8c…	2023-05-02 …	yona-hco7.n…	USAGE	QUERY	2023-05-02 …	shql-query-…	admin
1	{'app_name…	EV/11ede8c…	2023-05-02 …	yona-hco7.n…	USAGE	ADD	2023-05-02 …	shql-query-…	admin
1	{'duration': …	EV/11ede8c…	2023-05-02 …	yona-hco7.n…	USAGE	QUERY	2023-05-02 …	shql-query-…	admin
1	{'duration': …	EV/11ede8c…	2023-05-02 …	yona-hco7.n…	USAGE	QUERY	2023-05-02 …	shql-query-…	admin
1	{'app_name…	EV/11ede83…	2023-05-01 …	yona-hco7.n…	USAGE	OPEN	2023-05-01 …	shql-query-…	admin

Complex Queries

Example I

- The next query will show the Physical ethernet ports used by links in a specific site:

  site[.name contains "MIL"] | inventory_item | port[.type="R_PHYSICAL"] | link | port



SHQL

Saved Queries | Save | RUN

```
site[.name contains "MIL"] | inventory_item | port[.type="R_PHYSICAL"] | link | port
```

RESULTS (22)

Ethernet Port (4) | L3 Physical Port (18)

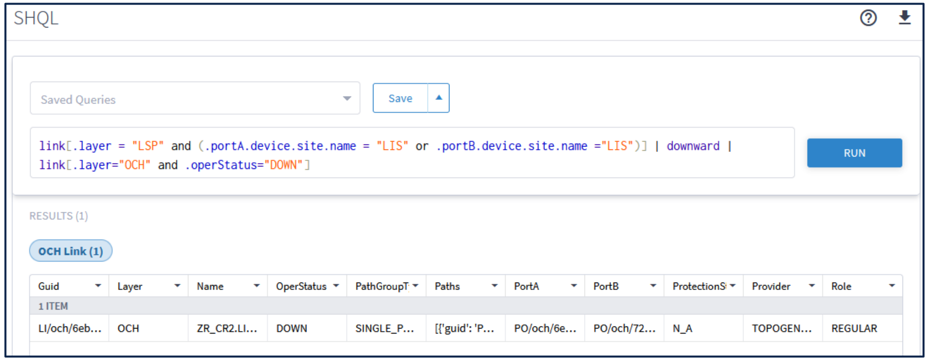
4 ITEMS

Guid	Type	AdminStatus	Device	Name	OperStatus	Parent	Provider	EthPortType	SpeedBps
PO/eth/0ba...	ETH	UP	IN/ROADM/t...	2-2-4	UP	IN/PORT/a1...	topogen-cie...	ETH_10G	10000000000
PO/eth/0ba...	ETH	UP	IN/ROADM/t...	2-5-1	UP	IN/PORT/ca...	topogen-cie...	ETH_10G	10000000000
PO/eth/0ba...	ETH	UP	IN/ROADM/t...	2-4-2	UP	IN/PORT/2af...	topogen-cie...	ETH_10G	10000000000
PO/eth/0ba...	ETH	UP	IN/ROADM/t...	2-3-3	UP	IN/PORT/90...	topogen-cie...	ETH_10G	10000000000

Example II

- Lisbon LSP down due to OCH failure

- link[.layer = "LSP" and (.portA.device.site.name = "LIS" or .portB.device.site.name ="LIS")] | downward | link[.layer="OCH" and .operStatus="DOWN"]


SHQL

Saved Queries

Save

```
link[.layer = "LSP" and (.portA.device.site.name = "LIS" or .portB.device.site.name ="LIS")] | downward |
link[.layer="OCH" and .operStatus="DOWN"]
```

RUN

RESULTS (1)

OCH Link (1)

Guid	Layer	Name	OperStatus	PathGroupT	Paths	PortA	PortB	ProtectionS	Provider	Role
1 ITEM										
LI/och/6eb...	OCH	ZR_CR2.LI...	DOWN	SINGLE_P...	[{'guid': 'P...	PO/och/6e...	PO/och/72...	N_A	TOPOGEN...	REGULAR

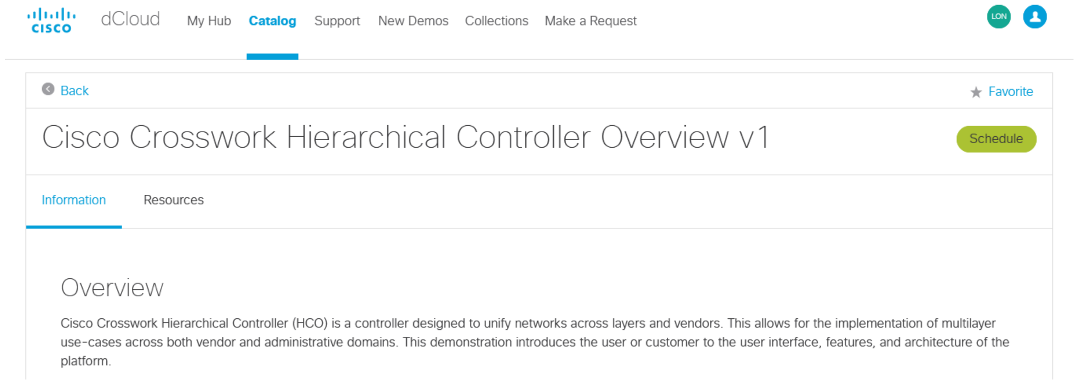
Demo and Examples

- SHQL App
- REST API
- CLI
- NSO

Some final words from Yona and Daniel

- This session has been focused on HCO 7 and the “read-only” query language SHQL. In HCO 8 provisioning/”write-to” will be enable via the service manager API

- For firsthand experience, make sure to book a dCloud lab



Cisco Crosswork Hierarchical Controller Overview v1

Overview

Cisco Crosswork Hierarchical Controller (HCO) is a controller designed to unify networks across layers and vendors. This allows for the implementation of multilayer use-cases across both vendor and administrative domains. This demonstration introduces the user or customer to the user interface, features, and architecture of the platform.

The bridge to possible