

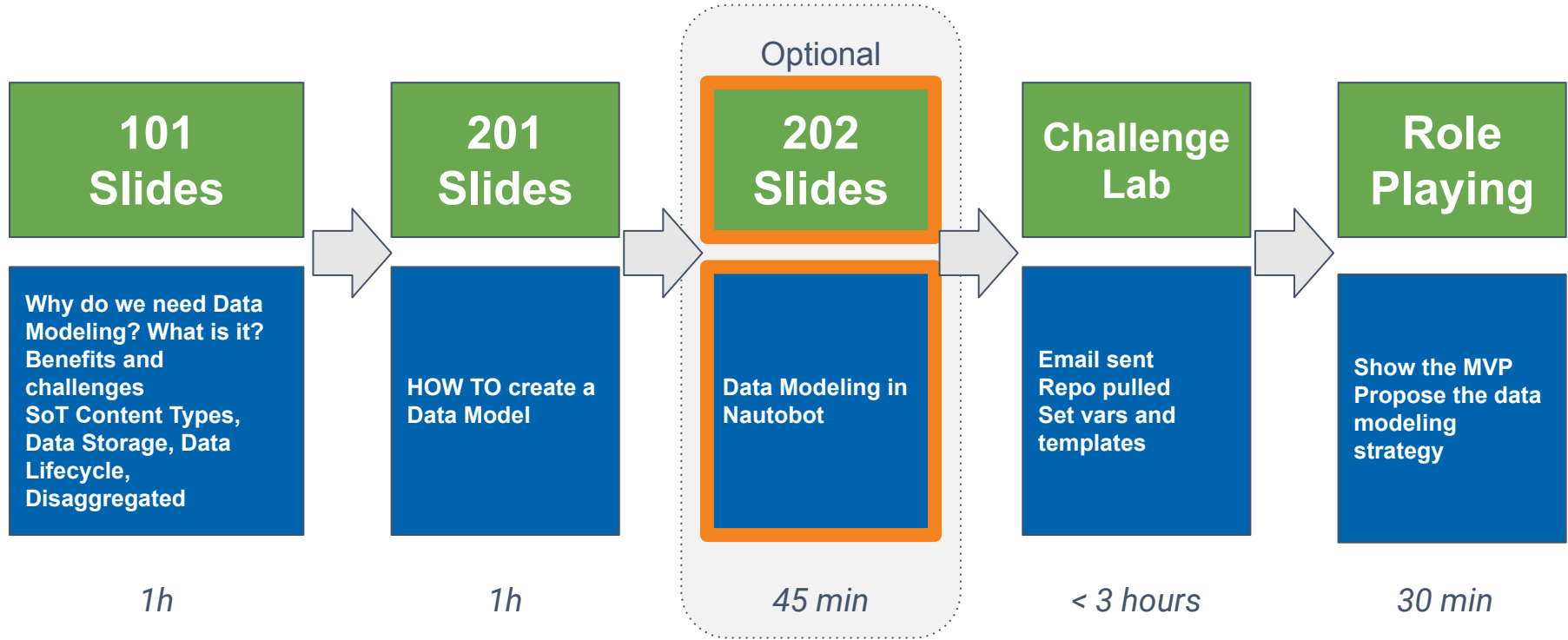
>>>network.toCode()

202 - Data Modeling

How to leverage Nautobot to represent
your Data Models

Aug 16th, 2022

>>> Data Modeling Training Plan



>>> Nautobot

Source of Truth and Network Automation Platform

<https://demo.nautobot.com>



nautobot

Widgets

Core View
Extensions

Navigation

Validators

Git

Jobs

REST API

GraphQL

Data Models

APIs & Business Logic

Web UI Views

Extensibility Features



Nautobot
Apps



>>> nautobot

Core

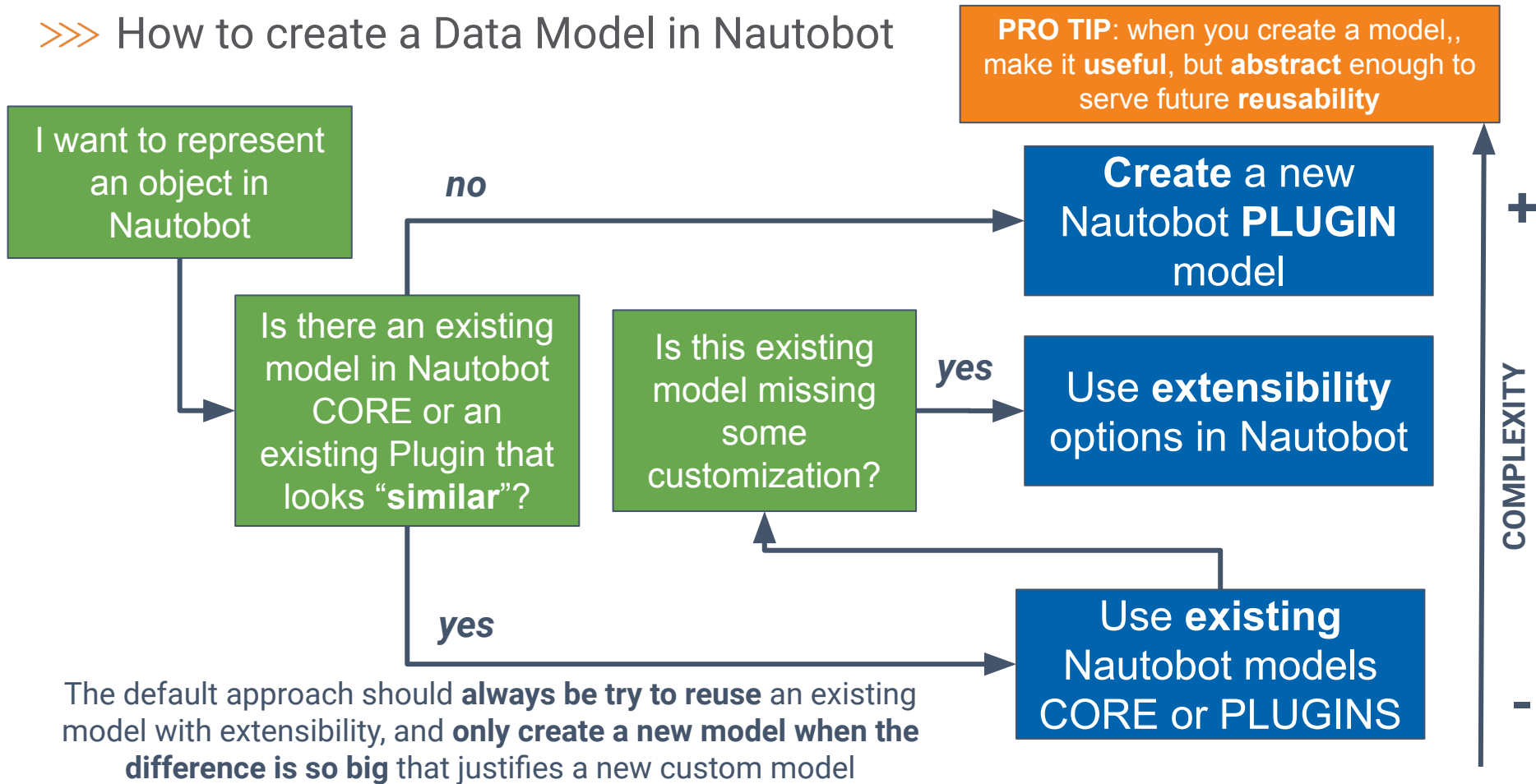
django

 PostgreSQL

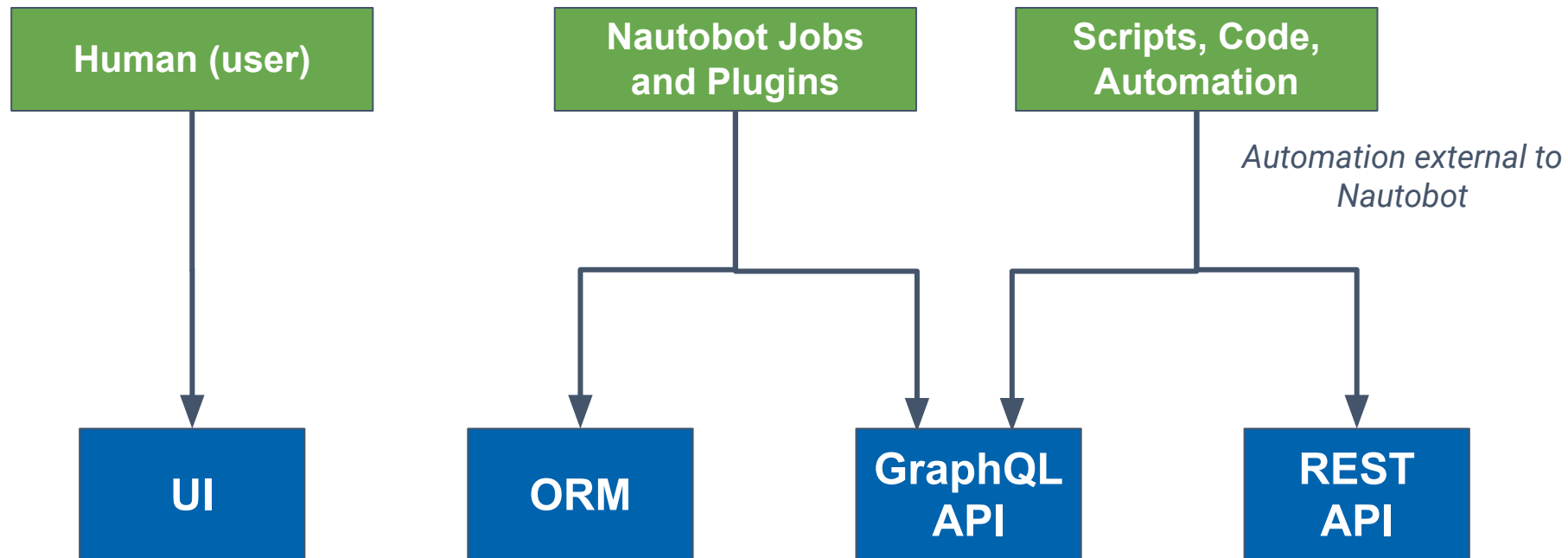
 MySQL

d01t

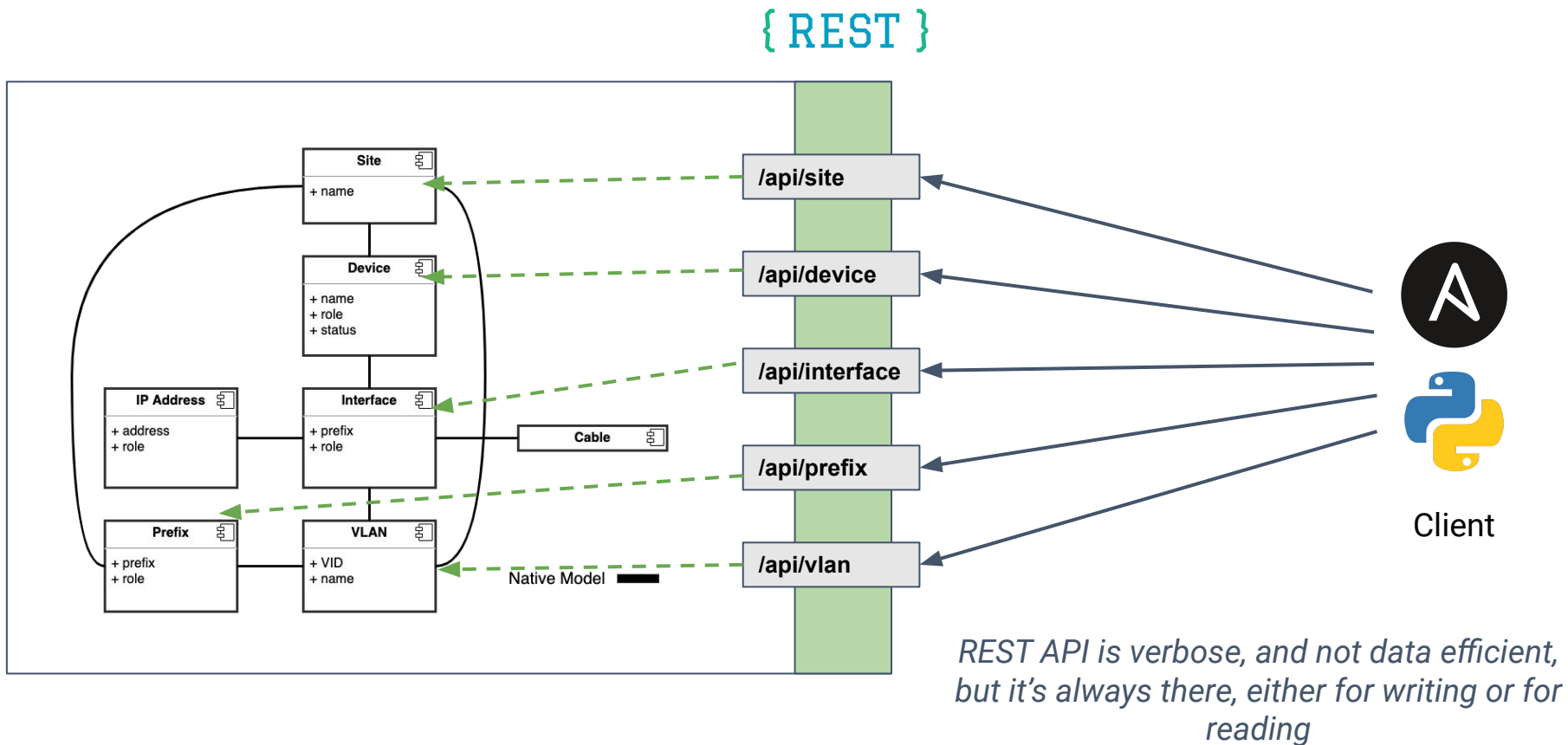
>>> How to create a Data Model in Nautobot



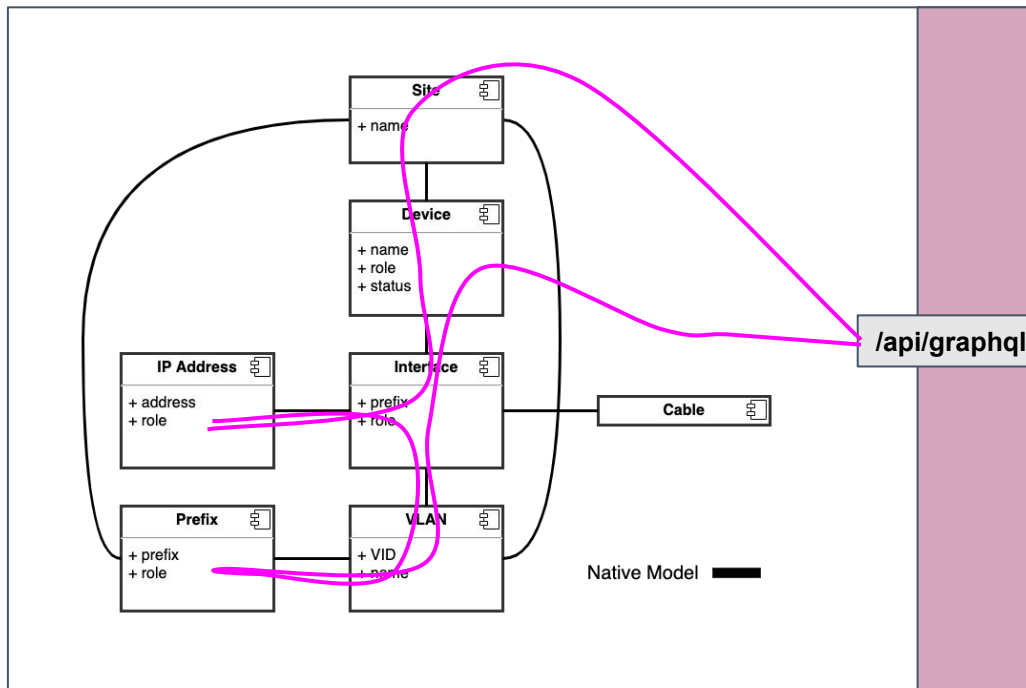
>>> How to consume/populate data in Nautobot



>>> Query multiple objects via the REST API



>>> GraphQL API



Client

*GraphQL helps to GET the data as you want it, without having to get everything and manipulate locally. Generally **preferred to get data***

>>> Interfaces - GraphQL

The screenshot displays the Nautobot web interface with the GraphQL endpoint active. The top navigation bar includes links for Organization, Devices, IPAM, Routing, Virtualization, Circuits, Power, Device Lifecycle, Secrets, Jobs, Extensibility, Security, Golden Config, and Plugins. A search bar and a user profile icon are also present.

The GraphQL editor shows a query to fetch details for a specific device:

```
1 query {  
2   devices(name: "bre01-dist-01") {  
3     name  
4     location: site {  
5       name  
6       slug  
7       region {  
8         name  
9       }  
10    }  
11    config_context  
12    interfaces {  
13      ip_addresses {  
14        id  
15      }  
16      name  
17      connected_interface {  
18        device {  
19          name  
20        }  
21      }  
22      connected_circuit_termination {  
23        circuit {  
24          cid  
25        }  
26      }  
27    }  
28  }  
29 }
```

The response is a JSON object:

```
{  
  "data": {  
    "devices": [  
      {  
        "name": "bre01-dist-01",  
        "location": {  
          "name": "BRE01",  
          "slug": "bre01",  
          "region": {  
            "name": "United States"  
          }  
        },  
        "config_context": {  
          "cdp": true,  
          "ntp": [  
            {  
              "ip": "10.1.1.1",  
              "prefer": false  
            },  
            {  
              "ip": "10.2.2.2",  
              "prefer": true  
            }  
          ],  
          "lldp": true,  
          "snmp": {  
            "host": [  
              {  
                "ip": "10.1.1.1",  
                "version": "2c",  
                "community": "networktocode"  
              }  
            ],  
            "contact": "John Smith",  
            "location": "Network to Code - NYC | NY",  
            "community": [  
              {  
                "name": "ntc-public",  
                "role": "RO"  
              },  
              {  
                "name": "ntc-private",  
                "role": "RW"  
              },  
              {  
                "name": "networktocode",  
                "role": "RO"  
              },  
              {  
                "name": "secure",  
                "role": "RW"  
              }  
            ]  
          }  
        }  
      }  
    ]  
  }  
}
```

On the right, the Documentation Explorer shows a search bar and a description: "A GraphQL schema provides a root type for each kind of operation." Below this, it lists "ROOT TYPES" and shows the selected "query: Query".

At the bottom, the "QUERY VARIABLES" section is visible, showing the variable `device` set to `"bre01-dist-01"`.



>>> Extending Nautobot Data Models

>>> Customizing Nautobot

...without coding

Nautobot offers easy extensibility to adapt it to specific use-cases

Computed Fields	Custom Fields	Webhooks / Jobhooks
Statuses	Custom Links	Export Templates
Relationships	Config Context	Git
Tags	Config Context Schema	GraphQL Queries

>>> Computed Fields

Extensibility -> Miscellaneous -> Computed Fields

Create new data field from the DB using Jinja2 templating

Add a new computed field

Computed field

Content Type

circuits | circuit

Label

complete-circuit-id

Name of the field as displayed to users

Slug

complete-circuit-id

URL-friendly unique shorthand

Description

Description

Template

{{ obj.provider.slug }}_{{ obj.cid }}

Jinja2 template code for field value

Fallback value

undefined

Fallback value to be used for the field in the case of a template rendering error.

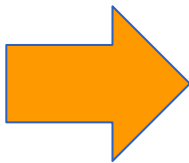
Weight

100

Create

Create and Add Another

Cancel



Circuits / AT&T / 123456789

123456789

Created Sept. 15, 2021 · Updated 0 minutes ago

Circuit

Change Log

Circuit

Status	Active
Provider	AT&T
Circuit ID	123456789
Type	Transit
Tenant	None
Install Date	—
Commit Rate	—
Description	—

Computed Fields

complete-circuit-id	att_123456789
---------------------	---------------

Useful to automatically generate attributes whose values follow some well_known pattern. E.g. router hostname characters 4&5 encode the floor number (computed_field_floor {{ obj.name[3:4] }}) on nyc11-rt01 == 11

>>> Statuses

Organization -> Statuses

Define your own Statuses for specific use-cases

Add a new status

Status

Name

Draining

Slug

draining

URL-friendly unique shorthand

Description

Description

Content Type(s)

× circuits | circuit

×

Color

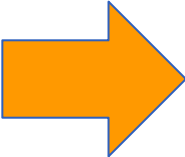
Purple

×

Create

Create and Add Another

Cancel



Circuits / AT&T / 123456789

123456789

Created Sept. 15, 2021 · Updated 0 minutes ago

Circuit

Change Log

Circuit

Status

Draining

Provider

AT&T

Circuit ID

123456789

Type

Transit

Tenant

None

Install Date

—

Commit Rate

—

Description

—

Use Status on any object to describe the state and **potentially** consider in scope or not, e.g remove **Decomm**

@networktoCode | Confidential

12

>>> network.toCode()

>>> Relationships

Extensibility -> Relationships

Establish new dependencies between Models

Editing relationship Prefix per Circuit

Relationship

Name Prefix per Circuit
Internal relationship name

Slug prefix_circuit
URL-friendly unique shorthand

Description Description

Type One to One

Source type circuits | circuit

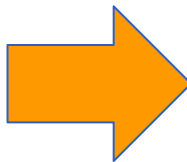
Source Label Source Label
Name of the relationship as displayed on the source object.

☐ **Hide for source object**
Hide this relationship on the source object.

Source filter None
Enter any filters for the source object in JSON format.

Destination type ipam | prefix

Destination Label Destination Label
Name of the relationship as displayed on the destination object.



Circuits / AT&T / 123456789

123456789

Created Sept. 15, 2021 · Updated 0 minutes ago

Circuit [Change Log](#)

Circuit

Status	Draining
Provider	AT&T
Circuit ID	123456789
Type	Transit
Tenant	None
Install Date	—
Commit Rate	—
Description	—

Computed Fields

complete-circuit-id	att_123456789
---------------------	---------------

Relationships

prefix	10.0.0.0/8
--------	------------

Relationships allow linking of objects to enforce business logic (not present already), such as assigning an IP Address to a Circuit

>>> Tags

Organization -> Tags

Enrich with metadata!

Add a new tag

Tag

Name

Low cost

Slug

low-cost

URL-friendly unique shorthand

Color

Light green

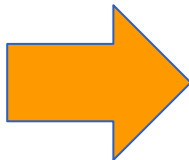
Description

Low cost Circuit

Create

Create and Add Another

Cancel



Circuits / AT&T / 123456789

123456789

Created Sept. 15, 2021 - Updated 0 minutes ago

Circuit

Change Log

Circuit

Status

Draining

Provider

AT&T

Circuit ID

123456789

Type

Transit

Tenant

None

Install Date

—

Commit Rate

—

Description

—

Computed Fields

complete-circuit-id

att_123456789

Relationships

prefix

10.0.0.0/8

Tags

Low cost

Tags are useful for adding things that are not mutually exclusive (there can be many), like saying "services on a device" and having voice/video/data (do not abuse! *Structure is not enforced*)

>>> Custom Fields

Admin portal -> Extras -> Custom Fields

What data do you need to store?

Add a new custom field

Custom Field

Slug

order_id

URL-friendly unique shorthand.

Type

Text

The type of value(s) allowed for this field.

Weight

100

Fields with higher weights appear lower in a form.

Label

Order ID

Name of the field as displayed to users (if not provided, the field's slug will be used.)

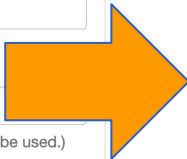
Description

None

Also used as the help text when editing models using this custom field.
Markdown syntax is supported.

☒ Required

If true, this field is required when creating new objects or editing an existing object.



Note: Remember that applies to all the object of a type

Add a new circuit

Circuit

Provider

Circuit ID

Circuit ID

Unique circuit ID

Type

Status

Date installed

YYYY-MM-DD

Commit rate (Kbps)

Commit rate (Kbps)

Committed rate

Description

Description

Tenancy

Tenant group

Tenant

Custom Fields

Order id

Order id

Adding a new attribute to extend current models/tables as needed without having to create a new model

>>> Custom Link

Extensibility -> Miscellaneous -> Custom Links

Do you need to inject a hyperlink?

Add a new custom link

Custom link

Content Types: circuits | circuit

Name: Grafana monitoring

Text: {{ obj.provider.slug }}_{{ obj.cid }}

URL: https://mygrafana_dashboard/{{ obj.provider.slug }}_{{ obj.cid }}

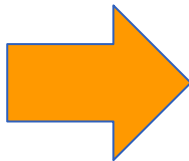
Weight: 100

Group name: Group name

Button class: Primary (blue)

☐ New window
Force link to open in a new window

Create Create and Add Another Cancel



Circuits / AT&T / 123456789

+ Clone Edit Delete

123456789

Created Sept. 15, 2021 · Updated 14 minutes ago

Circuit Change Log

att_123456789

Circuit

Status Draining

Provider AT&T

Circuit ID 123456789
https://mygrafana_dashboard/att_123456...

To enhance UI experience, you can create your own hyperlinks that will redirect you to external sites (i.e. external monitoring tool)

>>> Config Context

Extensibility -> Automation -> Config Context

Add a new config context

Config Context

Name:

Weight:

Description:

Schema:

Optional schema to validate the structure of the data

☒ Is active

Assignment

Regions:

Sites:

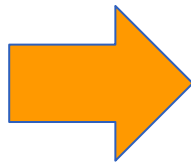
Data

```
{
  "dns_servers": ["1.1.1.1", "8.8.8.8", "9.9.9.9"]
}
```

Enter context data in JSON format.

Create Create and Add Another Cancel

Config Context offers a hierarchical approach to customize Device/VM objects by different filtering options. You can also take them from Git Datasource!



Devices / AMS01 / ams01-edge-01

ams01-edge-01

Created Sept. 7, 2021 · Updated 0 minutes ago

Device Interfaces 62 Console Ports 1 Power Ports 2 **Config Context** Change Log

Rendered Context JSON | YAML

```
{
  dns_servers:
    - 1.1.1.1
    - 8.8.8.8
    - 9.9.9.9
}
```

When variables get applied to many devices, but you do not need to manage per device, but rather per site/region/role, etc.

>>> Config Context Schema

Extensibility -> Automation -> Config Context Schemas

Enforce a correct data structure

Add a new config context schema

Config Context Schema

Name

Slug

URL-friendly unique shorthand

Description

Data Schema

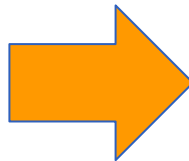
```
{  "type": "object",  "properties": {    "dns_servers": {      "type": "array",      "minItems": 2,      "maxItems": 2,      "items": {        "type": "string",        "format": "ipv4"      }    }  },  "additionalProperties": false}
```

Enter context data in JSON format.

Create

Create and Add Another

Cancel



Editing config context DNS servers

Config Context

Name

Weight

Description

Schema

Data

```
{  "dns_servers": [    "1.1.1.1",    "8.8.8.8",    "9.9.9.9"  ]}
```

Enter context data in JSON format.

- Validation using the JSON Schema DNS server schema failed.
- ['1.1.1.1', '8.8.8.8', '9.9.9.9'] is too long

Update

Cancel

Config Context is great but skips DB attribute validation. CC Schema solves this validation issue.

>>> Webhooks / Jobhooks

Extensibility -> Automation -> Webhooks

Webhooks triggers an external API call when an object type change happens. Jobhooks is the equivalent, but triggering Nautobot Jobs

Add a new webhook

Webhook

Name

Content Type(s)

☒ Enabled

☒ Type create
Call this webhook when a matching object is created.

☒ Type update
Call this webhook when a matching object is updated.

☒ Type delete
Call this webhook when a matching object is deleted.

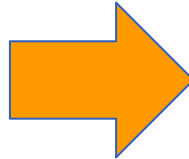
URL

A POST will be sent to this URL when the webhook is called.

HTTP method

HTTP content type

The complete list of official content types is available [here](#).



```
root@62b87b3334e3:/source# nc -l -p 8008
POST / HTTP/1.1
Host: localhost:8008
Accept-Encoding: identity
Content-Type: application/json
Content-Length: 1032
User-Agent: python-urllib3/1.26.4

{"event": "created", "timestamp": "2021-09-15 13:18:42.497319+00:00", "model": "circuit", "username": "admin", "request_id": "22c84540-1da4-415d-91bd-4e1be402f225", "data": {"id": "7fdaff08-aca4-418b-b71b-b5a01de1e7c", "url": "/api/circuits/circuits/7fdaff08-aca4-418b-b71b-b5a01de1e7c/", "cid": "new circuit id", "provider": {"id": "b37d5dfc-f9e7-459f-b7ff-b0a1fd0ee37", "url": "/api/circuits/providers/b37d5dfc-f9e7-459f-b7ff-b0a1fd0ee37/", "name": "ntt", "slug": "ntt", "display": "ntt"}, "type": {"id": "27751fa7-b8b6-47c3-b18e-e11dc66c8c7", "url": "/api/circuits/circuit-types/27751fa7-b8b6-47c3-b18e-e11dc66c8c7/", "name": "transit", "slug": "transit", "display": "transit"}, "status": {"value": "active", "label": "Active"}, "tenant": null, "install_date": null, "commit_rate": null, "description": "", "termination_a": null, "termination_z": null, "comments": "", "tag": [], "custom_fields": {"Order ID": "223123", "created": "2021-09-15", "last_updated": "2021-09-15T13:18:42.433903Z", "display": "new circuit id"}}
```

Great to kick off automation when data changes!

>>> Export Templates

Extensibility -> Automation -> Export Templates

Retrieve the data your way

CSV (by default) or custom

Add a new export template

Export template

Content Types: circuits | circuit

Name: Circuits report

Description: Description

Template code

```
{% for circuit in queryset %}
Circuit: {{ circuit.cid }}
Provider: {{ circuit.provider.name }}
Status: {{ circuit.status }}
{% endfor %}
```

The list of objects being exported is passed as a context variable named `queryset`.

MIME type: MIME type
Defaults to: text/plain

File extension: File extension
Extension to append to the rendered filename

Create Create and Add Another Cancel

```
nautobot_circuits -- Edited ✓
Circuit: 123456789
Provider: AT&T
Status: Draining

Circuit: ntt-104265404093023273
Provider: NTT
Status: Active

Circuit: ntt-104265404093069929464
Provider: NTT
Status: Active

Circuit: ntt-104539051754046505
Provider: NTT
Status: Active

Circuit: ntt-104539051754093161
Provider: NTT
Status: Active

Circuit: ntt-38874897548100649
Provider: NTT
Status: Active

Circuit: ntt-38874897548147305
Provider: NTT
Status: Active
```



Circuits

ID	Provider	Type	Status	Tenant	A Side	Z Side	Description
123456789	AT&T	Transit	Draining	—	—	—	—

Configure + Add Import Export

Default format: Circuits report

Export Templates are great to translate the Data Model representation to other formats for easier integration

>>> Git

Extensibility -> Data Sources -> Git Repositories

Use your favorite Git Svc to inject/extract data/config/templates

Git Repositories

[Configure](#)[Export](#)

Name	Remote URL	Branch	Provides	Sync Status
backups	https://github.com/nautobot/demo-gc-backups	main		N/A
configs	https://github.com/nautobot/demo-gc-generated-configs	main		N/A
data	https://github.com/nautobot/demo-git-datasource	main		N/A
templates	https://github.com/nautobot/demo-gc-templates	main		N/A

data

Created Sept. 7, 2021 · Updated 1 week, 1 day ago

[Git Repository](#) [Synchronization Status](#) [Change Log](#)

Repository Details

Remote URL	https://github.com/nautobot/demo-git-datasource
Branch	main (checked out locally at commit f18b001ed8ca28fd7c4a8a3e46ef9cf909e29a57)
Token	—

Tags

No tags assigned

Provided Data Types

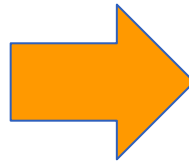
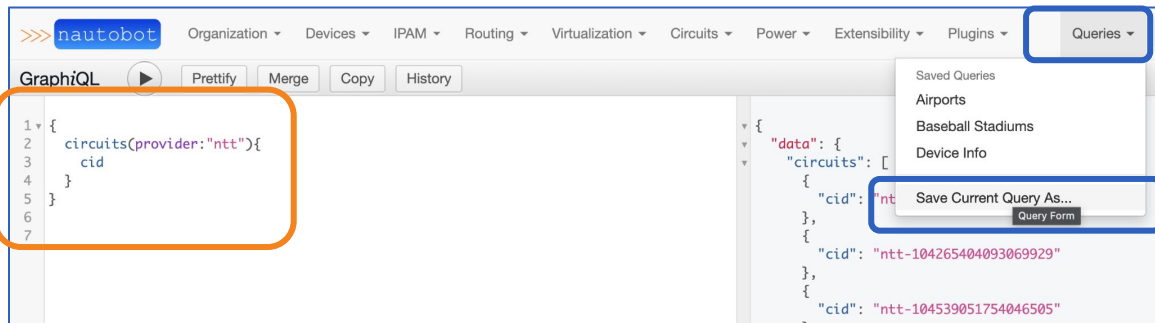
Config Context Schemas	×
Config Contexts	✓
Jobs	×
Export Templates	×
Backup Configs	×
Intended Configs	×
Jinja Templates	×

When some data for the model is in an external Git repository, it can be imported into Nautobot to combine

>>> GraphQL Queries

Extensibility -> Data Management -> GraphQL Queries

Nautobot remembers!



GraphQLQueries

Add a new:

Name

NTT circuits

Slug

ntt-circuits

URL-friendly unique shorthand

Query

```
{  
  circuits(provider:"ntt"){  
    cid  
  }  
}
```

Create

Cancel

REST API consumption can be not efficient enough when consuming the data model, GraphQL helps optimizing it



>>> Create **new** Nautobot data models

>>> New model in Nautobot

ams01-dist-01

Created May 4, 2022 · Updated 3 months, 3 weeks ago

Device Advanced Rear Ports 1 Inventory 1 Status LLDP Neighbors Config

Device	
Site	Netherlands / AMS01
Rack	None
Position	—
Tenant	Nautobot Airports
Device Type	Cisco Catalyst 6509-E (14U)
Serial Number	—
Asset Tag	—
Management	
Role	distribution
Platform	Cisco IOS

GET

/plugins/nautobot-device-lifecycle-mgmt/hardware/

▼

🔒

POST

/plugins/nautobot-device-lifecycle-mgmt/hardware/

▼

🔒

PUT

/plugins/nautobot-device-lifecycle-mgmt/hardware/

▼

🔒

PATCH

/plugins/nautobot-device-lifecycle-mgmt/hardware/

▼

🔒

DELETE

/plugins/nautobot-device-lifecycle-mgmt/hardware/

▼

🔒

GET

/plugins/nautobot-device-lifecycle-mgmt/hardware/{id}/

▼

🔒

PUT

/plugins/nautobot-device-lifecycle-mgmt/hardware/{id}/

▼

🔒

PATCH

/plugins/nautobot-device-lifecycle-mgmt/hardware/{id}/

▼

🔒

DELETE

/plugins/nautobot-device-lifecycle-mgmt/hardware/{id}/

▼

🔒

Hardware

Inventory

Support

Device Type: Catalyst 6509-E - End of support: 2025-10-31

➔

Software is valid

NameCisco IOS - 12.2(33)SX114

Device PlatformCisco IOS

Version12.2(33)SX114

Once the plugin model is available in Nautobot, it becomes a **first-class citizen**, and leverage all the extensibility options, even from other plugins

Can integrate with **RestAPI** or **GraphQL**. A custom model can integrate the same as a Core Model, right into **Core** pages or custom pages.

Has all the standard views that a Core model has

Bonus: **pynautobot** works automatically

Building a model takes time, but has all of the feature Core models do. Great for when use case is sufficiently complex

>>> What it takes?

develop ▾ nautobot-plugin-device-lifecycle

abates Corrected code formatting

..

- api
- graphql
- jobs
- migrations
- templates/nautobot_device_lifecycle_mgmt
- tests
- __init__.py
- admin.py
- choices.py
- const.py
- filters.py
- forms.py
- models.py**
- navigation.py
- signals.py
- software.py
- software_filters.py
- tables.py
- template_content.py
- urls.py
- utils.py
- views.py

```
@extras_features(
    "custom_fields",
    "custom_links",
    "custom_validators",
    "export_templates",
    "graphql",
    "relationships",
    "webhooks",
)

class ProviderLCM(OrganizationalModel):
    """ProviderLCM model for plugin."""

    # Set model columns
    name = models.CharField(max_length=100, unique=True)
    description = models.CharField(max_length=200, blank=True)
    physical_address = models.CharField(max_length=200, blank=True)
    country = models.CharField(max_length=3, blank=True)
    phone = models.CharField(max_length=20, blank=True)
    email = models.EmailField(blank=True, verbose_name="E-mail")
    portal_url = models.URLField(blank=True, verbose_name="Portal URL")
    comments = models.TextField(blank=True)

    csv_headers = [
        "name",
        "description",
        "physical_address",
        "country",
        "phone",
        "email",
        "portal_url",
        "comments",
    ]

    class Meta:
        """Meta attributes for the class."""

        verbose_name = "Vendor"
        ordering = ("name",)

    def __str__(self):
        """String representation of ProviderLCM."""
        return self.name

    def get_absolute_url(self):
        """Returns the Detail view for ProviderLCM models."""
        return reverse("plugins:nautobot_device_lifecycle_mgmt:provider_lcm_detail", args=[self.pk])
```

Move count related to utils.

Fix bug with soft images table not rendered if no images

Provider: Arista

Created May 4, 2022 · Updated 3 months, 3 weeks ago

Vendor [Change Log](#)

Vendor	
Name	Arista
Description	Secured Disposal of Devices
E-Mail	tac@arista.com
Phone	+1-408-547-5500
Address	5453 Great America Parkway, Santa Clara, CA 95054
Country	USA
Comments	—

REST API

```
{
  "count": 2,
  "next": null,
  "previous": null,
  "results": [
    {
      "id": "41d4fcb5-45db-4334-8b75-6d4c58e72265",
      "name": "Arista",
      "description": "Secured Disposal of Devices",
      "physical_address": "5453 Great America Parkway, Santa Clara, CA 95054",
      "phone": "+1-408-547-5500",
      "email": "tac@arista.com",
      "comments": "",
      "custom_fields": {},
      "display": "Arista"
    }
  ]
}
```

YAML

```
---
id: 41d4fcb5-45db-4334-8b75-6d4c58e72265
name: Arista
description: Secured Disposal of Devices
physical_address: 5453 Great America Parkway, Santa Clara, CA 95054
phone: "+1-408-547-5500"
email: tac@arista.com
comments: ''
custom_fields: {}
display: Arista
```



>>> Nautobot Data Population helpers

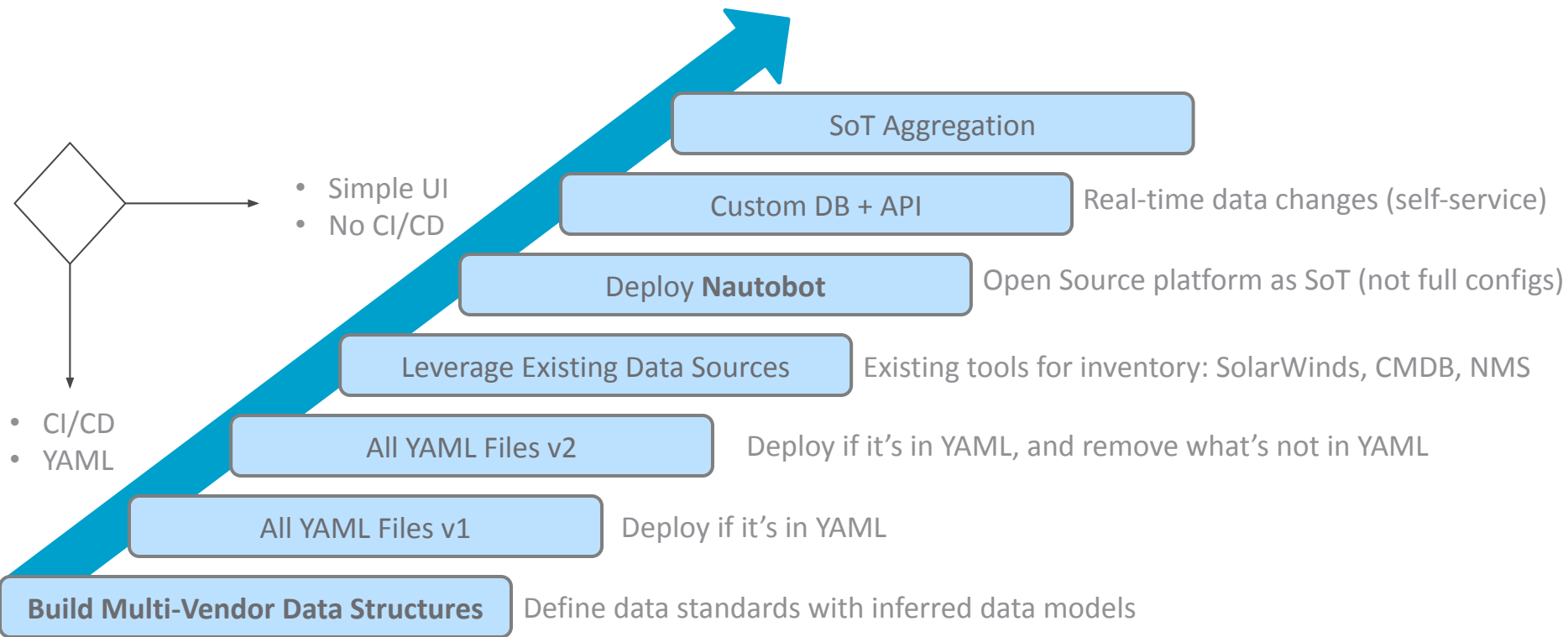
>>> Nautobot Plugins and Tools

Device Onboarding Plugin	Network Importer	SSoT Plugins
<ul style="list-style-type: none">• Automatically populates basic Nautobot Device information with minimum input• Initial Source of Truth data population, to be extended with Network Importer	<ul style="list-style-type: none">• Automatically populates information based on network state• Idempotently synchronizes Nautobot state from the network state• Raw and not-abstracted source of truth data population	<ul style="list-style-type: none">• Synchronizes Nautobot data from/to external applications (e.g. IPAM)• Used in case of many data domains serving the same or similar data



>>> SoT Adoption Strategy

>>> Sample Adoption Strategy



>>> Source of Truth Definition and Population

Devices

<input type="checkbox"/>	Name	Status	Tenant	Site	Rack	Role	Type
<input type="checkbox"/>	hou-bb-01	Active	—	HOU-01	HOU101	Router	Cisco cisco CSR1000V
<input type="checkbox"/>	hou-leaf-01	Active	—	HOU-01	HOU101	leaf	Arista vEOS
<input type="checkbox"/>	hou-leaf-02	Active	—	HOU-01	HOU102	leaf	Arista vEOS
<input type="checkbox"/>	hou-rtr-01	Active	—	HOU-01	HOU101	Router	Juniper VMX
<input type="checkbox"/>	hou-rtr-02	Active	—	HOU-01	HOU102	Router	Juniper VMX
<input type="checkbox"/>	hou-spine-01	Active	—	HOU-01	HOU101	spine	Arista vEOS
<input type="checkbox"/>	hou-spine-02	Active	—	HOU-01			
<input type="checkbox"/>	John-VMX-1	Active	—	LAS-01			

Interfaces

<input type="checkbox"/>	Name	LAG	Description	MTU	Mode	Cable	Connection
<input type="checkbox"/>	Ethernet1	—	—	1500	—	—	Not connected
	IP Address		Status/Role		VRF		
	10.255.0.61/30		Active		Global		
<input type="checkbox"/>	Ethernet2	—	—	1500	—	—	Not connected
	IP Address		Status/Role		VRF		
	10.255.0.70/30		Active		Global		
<input checked="" type="checkbox"/>	Ethernet3	—	—	1500	—	#326	hou-leaf-01
	IP Address		Status/Role		VRF		
	10.255.0.78/30		Active		Global		
<input checked="" type="checkbox"/>	Ethernet4	—	—	1500	—	#331	hou-leaf-02
	IP Address		Status/Role		VRF		
	10.255.0.110/30		Active		Global		
<input type="checkbox"/>	Ethernet5	—	—	1500	Access	—	Not connected

Rack HOU101

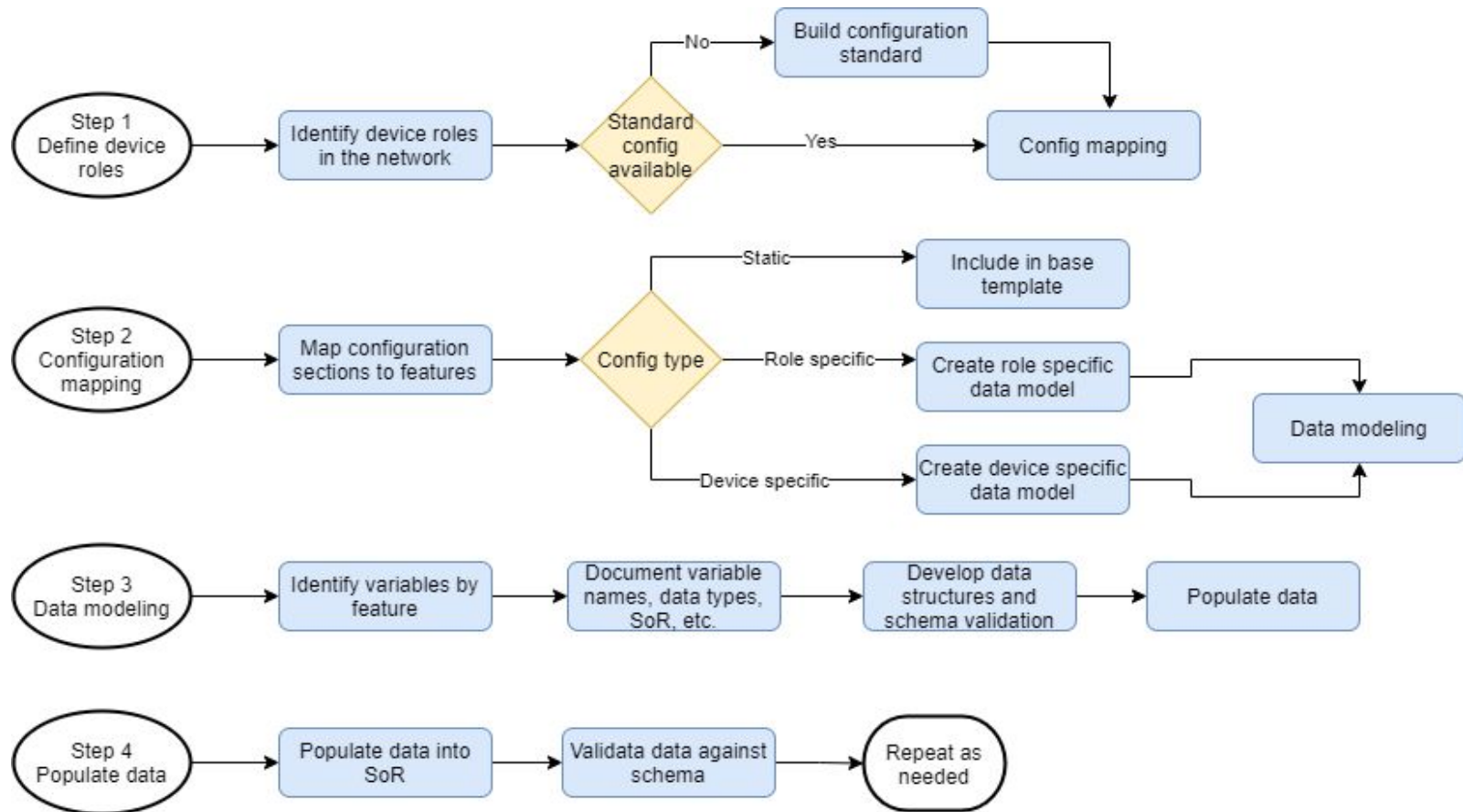
Created Sept. 10, 2019 • Updated 6 months, 3 weeks ago

Rack Change Log

Front
hou-leaf-01
hou-rtr-01
hou-spine-01
hou-leaf-01

1. **Establish** a central Source of Truth
2. **Extract** the **existing state** of your network
3. **Populate** the Source of Truth
4. From now on, **the network state is defined in the SoT**

>>> Modeling Workflow



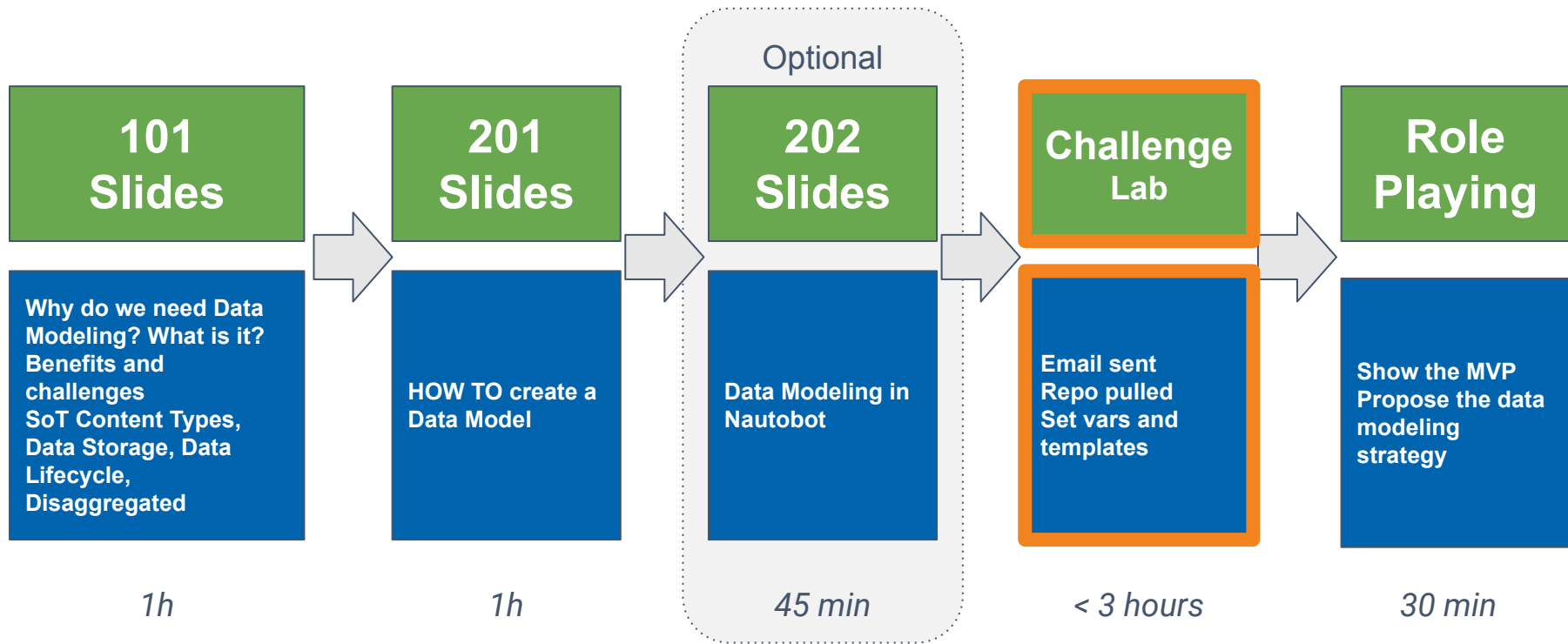
>>> Recommendations

- Build a strategy for your data
 - Where will the data be stored? How will it be accessed?
 - Define the system of record (SoR) for each component
 - Consider how current and future automation will use the data
 - Identify gaps
 - Document it!
- Start small
 - Focus on the system of record(s) that provides the most value for your immediate automation requirements
 - Identify a device role that has good config standardization and large impact
 - Model global device properties before attempting more complicated configurations/designs
 - Avoid trying to describe the entire design in data (i.e. if the config is standard for the design, it may belong in the template)
- Iterate
 - Accept that the first attempt at modeling won't cover all use cases
 - Improve on the existing data model as needed



>>> Next Step: Data Modeling Challenge

>>> Data Modeling Training Plan



>>>network.toCode()

Thank You