

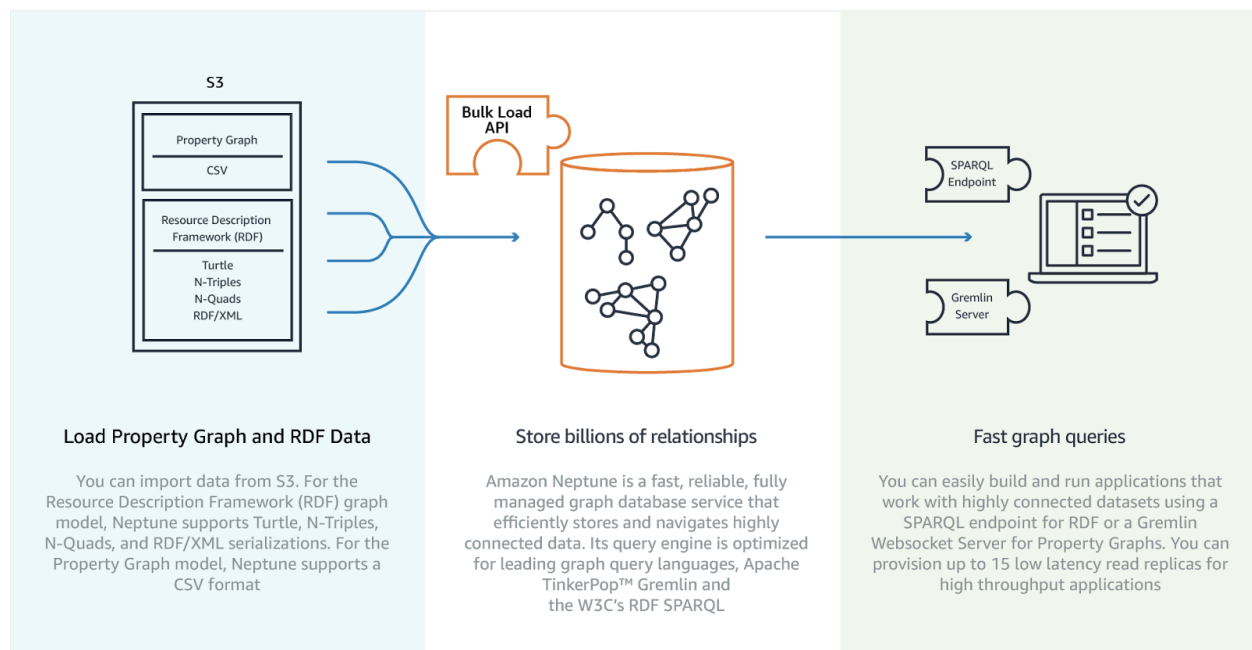


Thomson Reuters Graph Feed & Amazon Neptune

Tutorial on how to use Amazon Neptune as a graph-database alternative to explore Thomson Reuters Graph Feed.

Pratik Pandey
Jan 24th, 2018

End of November last year, Randall Hunt, an author and tech evangelist at Amazon, announced Amazon's foray into graph database with Neptune. Graph databases are dime a dozen, but Neptune stands out as being a fully-managed service with the ability to store billions of relationships of Property Graph and RDF data.



Neptune in a Nutshell

In this first blog post on Neptune, we will explore how we can export data from Thomson Reuters Knowledge Graph Feed and import it into Neptune. Thomson Reuters Knowledge Graph Feed is an API that delivers the Thomson Reuters Knowledge Graph. The raw size of Graph Feed export requires a graph analytics application to consume and query, and we'll see how to use Neptune for this.

[1. Export the content set from Graph Feed.](#)

[2. Setup Neptune](#)

[3. Create an S3 import bucket](#)

[4. Create Endpoint](#)

[5. Launch EC2 instance in one of the subnets inside the Neptune VPC](#)

[6. SSH into the EC2 instance and initiate load command](#)

[7. Validate](#)

By the end of this tutorial, you'll export content-set from Thomson Reuters Graph Feed, spin up your own Neptune instance, load it onto Neptune and query the data.

1. Export the content set from Graph Feed.

Let's go to DFGF to download a content set that you would like to import into Neptune. You can do that in few ways (Swagger Docs, Command-line with CURL, or from a client to the API), but at the end, it all boils down to a REST call to the Graph Feed APIs.

Here we'll be using the docs. First, use the `client_id` and `client_secret` to generate a short-lived token. If you do not have a `client_id` and secret, please reach out to Brian Rohan. Let's go to the [Graph Feed site](#).

THOMSON REUTERS
DATA FUSION™
GRAPH FEED

5LlvFHR7nd9ygnxQokTKAXkV09kR

This is the Data Fusion Graph Feed API documentation. All available API endpoints are documented below, including the ability to try any call. In order to try calls, you must enter an access token in the field above.

For help with using the Data Fusion Graph Feed API, please read the [step-by-step tutorial](#).

Updates to the endpoints will be documented on this page. For additional updates please see our [Release Notes](#).

1. Generate an Access Token

All interaction with the Data Fusion Graph Feed API requires an access token. You can generate an access token by entering your credentials here:

yiGOSp5EQAMrVJiCDCibkUTyrfYkxhLD



.....



Generate Token

The token can also be manually requested as follows (replacing `CLIENT_ID` and `CLIENT_SECRET` with your own credentials):

Use `/contentSet` to find a list of content-set that you're entitled to.

GET /contentSet Get the list of available content sets

Response Class (Status 200)

Model Example Value

```
[
  {
    "id": 0,
    "name": "string",
    "description": "string",
    "versions": [
      0
    ]
  }
]
```

Response Content Type application/json

Parameters

Parameter	Value	Description	Parameter Type	Data Type
entitled	true	Only show entitled content sets.	query	boolean

Grab the ID of the content set you want to export and supply that to /contentSet/{id}/download API endpoint. The Response here is a short-lived URL to download the content set. Go ahead and start the download while we setup Neptune.

GET /contentSet Get the list of available content sets

Response Class (Status 200)

Model Example Value

```
[
  {
    "id": 0,
    "name": "string",
    "description": "string",
    "versions": [
      0
    ]
  }
]
```

Response Content Type application/json

Parameters

Parameter	Value	Description	Parameter Type	Data Type
entitled	<input type="text" value="true"/>	Only show entitled content sets.	query	boolean

2. Setup Neptune

At this time of this writing, Neptune is only available in the us-east-1 region with a request for a preview, which you can do here.

<https://pages.awscloud.com/NeptunePreview.html>. Once you have access to Neptune you can setup Neptune service in AWS (<https://yukon.aws.amazon.com/rds/gdb?region=us-east-1>). Notice that this is not the regular console URL for AWS. Once you login to the yukon.aws console, setting up Neptune is a quick two-step process. First specify database details.

Specify DB details

Instance specifications

Estimate your monthly costs for the DB Instance using the [AWS Simple Monthly Calculator](#)

DB engine

neptune

DB instance class [info](#)

db.r4.4xlarge — 16 vCPU, 122 GiB RAM

db.r4.4xlarge — 16 vCPU, 122 GiB RAM

db.r4.8xlarge — 32 vCPU, 244 GiB RAM

Settings

DB instance identifier [info](#)

Specify a name that is unique for all DB instances owned by your AWS account in the current region.

dfgf-to-neptune

DB instance identifier is case insensitive, but stored as all lower-case, as in "mydbinstance".

Constraints:

- Must contain from 1 to 63 alphanumeric characters or hyphens.
- First character must be a letter.
- Cannot end with a hyphen or contain two consecutive hyphens.

Cancel

Next

Second in the *Configure Advanced Settings* make selection of your choices on VPC, backups, security etc. I went with creating a new VPC and choose defaults for other choices.

Step 1
[Specify DB details](#)

Step 2
Configure advanced settings

Configure advanced settings

Network & Security

Virtual Private Cloud (VPC) [info](#)

VPC defines the virtual networking environment for this DB instance.

Create new VPC



Only VPCs with a corresponding DB subnet group are listed.

Subnet group [info](#)

DB subnet group that defines which subnets and IP ranges the DB instance can use in the VPC you selected.

Create new DB Subnet Group

Availability zone [info](#)

No preference

VPC security groups

Security groups have rules authorizing connections from all the EC2 instances and devices that need to access the DB instance.

- ☒ Create new VPC security group
- ☐ Select existing VPC security groups

Once you configure everything and hit submit AWS will take a few minutes to instantiate the cluster. If everything is successful you will see the VPC and all the subnets Neptune cluster created.

The screenshot shows the AWS Management Console interface. On the left is a navigation menu with options like Clusters, Instances, Snapshots, Security groups, Subnet groups (highlighted), Parameter groups, Events, and Event subscriptions. The main content area shows the 'Neptune' console page, specifically the 'Subnet groups' section for 'default-vpc-52310e2a'. Below the title, there's a 'Subnet group details' section with fields for VPC ID (dfgf-neptune-vpc (vpc-52310e2a)), ARN (arn:aws:rds:us-east-1:217880599903:subgrp:default-vpc-52310e2a), and Description (Created from the RDS Management Console). Below this is a 'Subnets (6)' section containing a table with 6 subnets.

Availability zone	Subnet ID	CIDR block
us-east-1b	subnet-ae88ade5	172.30.1.0/24
us-east-1f	subnet-0bc23704	172.30.5.0/24
us-east-1c	subnet-868f1cdb	172.30.2.0/24
us-east-1e	subnet-567ae579	172.30.4.0/24
us-east-1a	subnet-0c204e33	172.30.0.0/24
us-east-1d	subnet-6cc79708	172.30.3.0/24

Now we will need to do few things in AWS so set up a pipeline to bulk import data into Neptune instance. Although in this tutorial we're focused on importing data from Graph Feed, this setup is required for bulk upload from any data source into Neptune.

3. Create an S3 import bucket

Create a new S3 bucket. Make sure you pick US East region. Once you have the bucket setup upload the file you downloaded from GraphFeed.

4. Create Endpoint

Creating an endpoint simplifies access to S3 resources from within a VPC and provides a secure connection to S3 that does not require a gateway or NAT instances. You can access endpoint configuration under VPC services. Make sure you choose **com.amazonaws.us-east.s3** for service and the Neptune VPC you created earlier.

Create Endpoint

A VPC endpoint allows you to securely connect your VPC to another service.

An interface endpoint is powered by [PrivateLink](#), and uses an elastic network interface (ENI) as an entry point for traffic destined to the service.

A gateway endpoint serves as a target for a route in your route table for traffic destined for the service.

- Service category**
- ☒ AWS services
 - ☐ Find service by name
 - ☐ Your AWS Marketplace services


Service Name com.amazonaws.us-east-1.s3 ⓘ

s3

	Service Name	Owner	Type
<input type="radio"/>	com.amazonaws.us-east-1.dynamodb	amazon	Gateway
<input type="radio"/>	com.amazonaws.us-east-1.ec2	amazon	Interface
<input type="radio"/>	com.amazonaws.us-east-1.ec2messages	amazon	Interface
<input type="radio"/>	com.amazonaws.us-east-1.elasticloadbala...	amazon	Interface
<input type="radio"/>	com.amazonaws.us-east-1.kinesis-streams	amazon	Interface
<input checked="" type="radio"/>	com.amazonaws.us-east-1.s3	amazon	Gateway
<input type="radio"/>	com.amazonaws.us-east-1.servicecatalog	amazon	Interface
<input type="radio"/>	com.amazonaws.us-east-1.ssm	amazon	Interface

VPC* vpc-52310e2a ⓘ ⓘ

Configure route tables

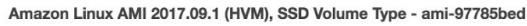
 Filter by attributes			
vpc-80aee2e7	10.50.0.0/20	available	DataFusion-Staging
vpc-59266b32	10.60.160.0/20	available	
vpc-24692e43	10.0.0.0/16	available	eb-vpc
vpc-52310e2a	172.30.0.0/16	available	dfgf-neptune-vpc
vpc-7ae8761d	10.194.56.0/25	available	graphfeed
vpc-2698745e	10.80.0.0/20	available	dev-main
vpc-3833c540	10.70.0.0/20	available	prod

target with this endpoints' ID

point.

5. Launch EC2 instance in one of the subnets inside the Neptune VPC

Create an EC2 instance in one of the subnets inside the Neptune VPC. The only thing to make sure here is that you pick the VPC that Neptune created.

[Edit AMI](#)

The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, Ruby, Perl, and Java. The repositories include Docker, PHP, MySQL, PostgreSQL, and other packages.

Root Device Type: ebs Virtualization type: hvm

[Edit instance type](#)

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
t2.micro	Variable	1	1	EBS only	-	Low to Moderate

[Edit security groups](#)[Edit instance details](#)

Number of instances	1	Purchasing option	On demand
Network	vpc-52310e2a		
Subnet	subnet-0c204e33		
EBS-optimized	No		
Monitoring	No		
Termination protection	No		
Shutdown behavior	Stop		
IAM role	None		
Tenancy	default		
T2 Unlimited	Disabled		
Host ID			
Affinity	Off		
Kernel ID	Use default		
RAM disk ID	Use default		
User data			
Assian Public IP	Use subnet setting (Enable)		

Cancel Previous **Launch**

Now it is just a matter of running a CURL with a POST command in the ec2 instance to load data into Neptune. Make sure to replace your own Neptune Endpoint, source, format, and credentials. If all goes well, you'll see a HTTP 200 response and a loadId in the payload.

```
[ec2-user@ip-172-30-0-119 ~]$ ./neptune-loader.sh
{
  "status" : "200 OK",
  "payload" : {
    "loadId" : "91c29b70-fad5-4474-81ed-941f96c022de"
  }
}
```


7. Validate

Once you see the response with loadId, you can use that to check the status of the import with a GET call to Neptune endpoint. As you see here in the following screenshot, there were 7,225,094 total records in this content set.

```
[ec2-user@ip-172-30-0-119 ~]$ curl -X GET http://neptune.cluster-cubddbui6ju.us-east-1-beta-2/loader/91c29b70-fad5-4474-81ed-941f96c022de
{
  "status" : "200 OK",
  "payload" : {
    "feedCount" : [
      {
        "LOAD_COMPLETED" : 1
      }
    ],
    "overallStatus" : {
      "fullUri" : "https://s3.amazonaws.com/graphfeed-neptune/55.cache.txt",
      "runNumber" : 1,
      "retryNumber" : 0,
      "status" : "LOAD_COMPLETED",
      "totalTimeSpent" : 78,
      "totalRecords" : 7225094,
      "totalDuplicates" : 0,
      "parsingErrors" : 0,
      "datatypeMismatchErrors" : 0,
      "insertErrors" : 0
    }
  }
}
```

You can now use SPARQL to query your data with an HTTP POST.

```
~/dev ssh -i "neptune-uploader.pem" ec2-user@ec2-54-84-79-122.compute-1.amazonaws.com
Warning: Identity file neptune-uploader.pem not accessible: No such file or directory.
Last login: Mon Jan 22 16:42:18 2018 from 70.123.51.50

 _ | _ | _ )
 _ | ( _ /   Amazon Linux AMI
  _|\_|_|_|

https://aws.amazon.com/amazon-linux-ami/2017.09-release-notes/
2 package(s) needed for security, out of 2 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-30-0-119 ~]$ curl -X POST --data-binary 'query=select ?s ?p ?o where {?s ?p ?o} limit 10' http://neptune.cluster-cubddbui6ju.us-east-1-beta.rds.amazonaws.com:8182/sparql
{
  "head" : {
    "vars" : [ "s", "p", "o" ]
  },
  "results" : {
    "bindings" : [ {
      "s" : {
        "type" : "uri",
        "value" : "http://data.thomsonreuters.com/sc/snippet/4295859594_5037971263_141_134_nL5N1FN76X-2017-02-07"
      },
      "p" : {
        "type" : "uri",
        "value" : "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"
      },
      "o" : {
        "type" : "uri",
        "value" : "http://ontology.thomsonreuters.com/supplyChain#Snippet"
      }
    }
  ]
}
```