Amazon Neptune: User Guide
Copyright © 2018 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.
# Table of Contents

What is Neptune? ............................................................................................................................ 1  
Amazon Neptune Features ............................................................................................................... 2  
  Performance and Scalability ................................................................................................. 2  
  High Availability and Durability ........................................................................................ 2  
  Support for Open Graph APIs ............................................................................................ 2  
  Enhanced Data Security .................................................................................................... 2  
  Fully Managed Service .................................................................................................. 2  
What is a Graph Database? ................................................................................................................. 3  
Graph Database Uses ...................................................................................................................... 4  
Graph Queries and Traversals ............................................................................................................ 7  
Quick Start ...................................................................................................................................... 10  
  Prerequisites .................................................................................................................... 10  
  Launch the Amazon Neptune CloudFormation Stack .................................................. 11  
Accessing the Neptune Graph ......................................................................................................... 12  
  Gremlin ........................................................................................................................... 12  
  RDF / SPARQL ............................................................................................................. 15  
Getting Started .............................................................................................................................. 17  
  Setting Up ......................................................................................................................... 17  
  Neptune VPC Requirements ............................................................................................ 17  
  Creating a Security Group to Provide Access to the Neptune DB Instance in the VPC ... 19  
  Launching a DB Cluster ...................................................................................................... 20  
  Launch a Neptune DB Cluster Using the Console ......................................................... 20  
  Creating a Neptune Replica Using the Console .............................................................. 22  
Launching an EC2 Instance ............................................................................................................ 25  
Gremlin ........................................................................................................................................... 26  
  Neptune Gremlin Implementation Differences ............................................................ 27  
  Gremlin Console ................................................................................................................ 34  
  HTTP REST ..................................................................................................................... 36  
  Java .................................................................................................................................. 37  
  Python .............................................................................................................................. 40  
  .NET ................................................................................................................................. 41  
  Node.js ............................................................................................................................ 42  
  Gremlin HTTP and WebSocket API ................................................................................ 44  
  Next Steps ....................................................................................................................... 44  
SPARQL .......................................................................................................................................... 44  
  RDF4J Console ................................................................................................................ 45  
  HTTP REST ..................................................................................................................... 47  
  Java .................................................................................................................................. 47  
  RDF4J Workbench ......................................................................................................... 50  
  SPARQL HTTP API ........................................................................................................ 51  
  Query Status and Cancellation ....................................................................................... 53  
  Next Steps ....................................................................................................................... 55  
Cluster Status ............................................................................................................................. 55  
Neptune Service Errors ............................................................................................................... 55  
  Error Format .................................................................................................................... 56  
  Error Messages and Codes .............................................................................................. 56  
Loading Data into Neptune ............................................................................................................ 59  
Prerequisites: IAM and Amazon S3 ............................................................................................ 60  
  Creating an IAM Role to Access Amazon S3 ............................................................... 60  
  Adding the IAM Role to a Cluster .................................................................................. 61  
  Creating the Amazon S3 VPC Endpoint ...................................................................... 61  
Load Data Formats ......................................................................................................................... 62
<table>
<thead>
<tr>
<th>Data Types</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>AvailabilityZone</td>
<td>301</td>
</tr>
<tr>
<td>CharacterSet</td>
<td>302</td>
</tr>
<tr>
<td>CloudwatchLogsExportConfiguration</td>
<td>303</td>
</tr>
<tr>
<td>DBCluster</td>
<td>304</td>
</tr>
<tr>
<td>DBClusterMember</td>
<td>309</td>
</tr>
<tr>
<td>DBClusterOptionGroupStatus</td>
<td>310</td>
</tr>
<tr>
<td>DBClusterParameterGroup</td>
<td>311</td>
</tr>
<tr>
<td>DBClusterRole</td>
<td>312</td>
</tr>
<tr>
<td>DBClusterSnapshot</td>
<td>313</td>
</tr>
<tr>
<td>DBClusterSnapshotAttribute</td>
<td>316</td>
</tr>
<tr>
<td>DBClusterSnapshotAttributesResult</td>
<td>317</td>
</tr>
<tr>
<td>DBEngineVersion</td>
<td>318</td>
</tr>
<tr>
<td>DBInstance</td>
<td>320</td>
</tr>
<tr>
<td>DBInstanceStatusInfo</td>
<td>327</td>
</tr>
<tr>
<td>DBParameterGroup</td>
<td>328</td>
</tr>
<tr>
<td>DBParameterGroupStatus</td>
<td>329</td>
</tr>
<tr>
<td>DBSecurityGroupMembership</td>
<td>330</td>
</tr>
<tr>
<td>DBSubnetGroup</td>
<td>331</td>
</tr>
<tr>
<td>DomainMembership</td>
<td>333</td>
</tr>
<tr>
<td>DoubleRange</td>
<td>334</td>
</tr>
<tr>
<td>Endpoint</td>
<td>335</td>
</tr>
<tr>
<td>EngineDefaults</td>
<td>336</td>
</tr>
<tr>
<td>Event</td>
<td>337</td>
</tr>
<tr>
<td>EventCategoriesMap</td>
<td>339</td>
</tr>
<tr>
<td>EventSubscription</td>
<td>340</td>
</tr>
<tr>
<td>Filter</td>
<td>342</td>
</tr>
<tr>
<td>OptionGroupMembership</td>
<td>343</td>
</tr>
<tr>
<td>OrderableDBInstanceOption</td>
<td>344</td>
</tr>
<tr>
<td>Parameter</td>
<td>347</td>
</tr>
<tr>
<td>PendingCloudwatchLogsExports</td>
<td>349</td>
</tr>
<tr>
<td>PendingMaintenanceAction</td>
<td>350</td>
</tr>
<tr>
<td>PendingModifiedValues</td>
<td>352</td>
</tr>
<tr>
<td>Range</td>
<td>355</td>
</tr>
<tr>
<td>ResourcePendingMaintenanceActions</td>
<td>356</td>
</tr>
<tr>
<td>Subnet</td>
<td>357</td>
</tr>
<tr>
<td>Tag</td>
<td>358</td>
</tr>
<tr>
<td>Timezone</td>
<td>359</td>
</tr>
</tbody>
</table>
What Is Amazon Neptune?

Amazon Neptune is a fast, reliable, fully managed graph database service that makes it easy to build and run applications that work with highly connected datasets. The core of Neptune is a purpose-built, high-performance graph database engine that is optimized for storing billions of relationships and querying the graph with milliseconds latency. Neptune supports the popular graph query languages Apache TinkerPop Gremlin and W3C’s SPARQL, allowing you to build queries that efficiently navigate highly connected datasets. Neptune powers graph use cases such as recommendation engines, fraud detection, knowledge graphs, drug discovery, and network security.

Neptune is highly available, with read replicas, point-in-time recovery, continuous backup to Amazon S3, and replication across Availability Zones. Neptune provides data security features, with support for encryption at rest and in transit. Neptune is fully managed, so you no longer need to worry about database management tasks like hardware provisioning, software patching, setup, configuration, or backups.

To learn about using Amazon Neptune, we recommend that you start with the following sections:

- What Is a Graph Database? (p. 3)
- Amazon Neptune Quick Start Using AWS CloudFormation (p. 10)
- Getting Started with Neptune (p. 17)

Supports Open Graph APIs

Amazon Neptune supports open graph APIs for both Gremlin and SPARQL, and it provides high performance for both of these graph models and their query languages. You can choose the Property Graph (PG) model and its open source query language, Apache TinkerPop Gremlin graph traversal language, or you can use the W3C standard Resource Description Framework (RDF) model and its standard SPARQL Query Language.

Highly Secure

Neptune provides multiple levels of security for your database, including network isolation using Amazon VPC, encryption at rest using keys that you create and control through AWS Key Management Service (AWS KMS), and encryption of data in transit using Transport Layer Security (TLS). On an encrypted Neptune instance, data in the underlying storage is encrypted, as are the automated backups, snapshots, and replicas in the same cluster.

Fully Managed

With Amazon Neptune, you don't have to worry about database management tasks like hardware provisioning, software patching, setup, configuration, or backups.

You can use Neptune to create sophisticated, interactive graph applications that can query billions of relationships in milliseconds. SQL queries for highly connected data are complex and hard to tune for performance. Instead, Neptune allows you to use the popular graph query languages TinkerPop Gremlin and SPARQL to execute powerful queries that are easy to write and perform well on connected data. This significantly reduces code complexity and enables you to more quickly create applications that process relationships.

Neptune is designed to offer greater than 99.99 percent availability. It increases database performance and availability by tightly integrating the database engine with an SSD-backed virtualized storage layer that is built for database workloads. Neptune storage is fault-tolerant and self-healing, and disk failures are repaired in the background without loss of database availability. Neptune automatically detects
database crashes and restarts without the need for crash recovery or rebuilding the database cache. If the entire instance fails, Neptune automatically fails over to one of up to 15 read replicas.

Amazon Neptune Features

Neptune provides the following basic features and capabilities.

Performance and Scalability

Amazon Neptune is a high-performance graph database service that is optimized for processing graph queries. Neptune supports up to 15 low-latency read replicas across three Availability Zones to scale read capacity and execute more than 100,000 graph queries per second. You can easily scale your database deployment up and down from smaller to larger instance types as your needs change.

High Availability and Durability

Neptune is highly available and durable and is designed to provide greater than 99.99 percent availability. It features fault-tolerant and self-healing storage built for the cloud that replicates six copies of your data across three Availability Zones. Neptune continuously backs up your data to Amazon S3 and transparently recovers from physical storage failures. For high availability, instance failover typically takes less than 30 seconds.

Support for Open Graph APIs

Neptune supports open graph APIs for both Gremlin and SPARQL, and it provides high performance for both of these graph models and their query languages. You can choose the PG model and its open source query language TinkerPop Gremlin, or the RDF model and its standard query language SPARQL.

Enhanced Data Security

Amazon Neptune provides multiple levels of security for your database, including network isolation using Amazon VPC, encryption at rest using keys that you create and control through AWS KMS, and encryption of data in transit using TLS. On an encrypted Neptune instance, data in the underlying storage is encrypted, as are the automated backups, snapshots, and replicas in the same cluster.

Fully Managed Service

You don't have to worry about database management tasks like hardware provisioning, software patching, setup, configuration, or backups. Neptune automatically and continuously monitors and backs up your database to Amazon S3, enabling granular point-in-time recovery.
What Is a Graph Database?

Topics
- Graph Database Uses (p. 4)
- Graph Queries and Traversals (p. 7)

Graph databases like Amazon Neptune are purpose-built to store and navigate relationships. Graph databases have advantages over relational databases for certain use cases—including social networking, recommendation engines, and fraud detection—when you want to create relationships between data and quickly query these relationships. There are a number of challenges to building these types of applications using a relational database. It requires you to have multiple tables with multiple foreign keys. The SQL queries to navigate this data require nested queries and complex joins that quickly become unwieldy. And the queries don't perform well as your data size grows over time.

Neptune uses graph structures such as nodes (data entities), edges (relationships), and properties to represent and store data. The relationships are stored as first-order citizens of the data model. This condition allows data in nodes to be directly linked, dramatically improving the performance of queries that navigate relationships in the data. The interactive performance at scale in Neptune effectively enables a broad set of graph use cases.

A graph in a graph database can be traversed along specific edge types, or across the entire graph.

Graph databases can represent how entities relate by using actions, ownership, parentage, and so on. Whenever connections or relationships between entities are at the core of the data that you’re trying to model, a graph database is a natural choice. Therefore, graph databases are useful for modeling and querying social networks, business relationships, dependencies, shipping movements, and similar items.

You can use edges to show typed relationships between entities (also called vertices or nodes). Edges can describe parent-child relationships, actions, product recommendations, purchases, and so on. A relationship, or edge, is a connection between two vertices that always has a start node, end node, type, and direction.

An example of a common use case that is suited to a graph is social networking data. Amazon Neptune can quickly and easily process large sets of user profiles and interactions to build social networking applications. Neptune enables highly interactive graph queries with high throughput to bring social features into your applications. For example, suppose that you want to build a social feed into your application. You can use Neptune to provide results that prioritize showing your users the latest updates from their family, from friends whose updates they “Like,” and from friends who live close to them.

Following is an example of a social network graph.
This example models a group of friends and their hobbies as a graph. A simple traversal of this graph can tell you what Justin’s friends like.

Graph Database Uses

Graph databases are useful for connected, contextual, relationship-driven data. An example is modeling social media data, as shown in the previous section. Other examples include recommendation engines, driving directions (route finding), logistics, diagnostics, and scientific data analysis in fields like neuroscience.

Fraud Detection

Another use case for graph databases is detecting fraud. For example, you can track credit card purchases and purchase locations to detect uncharacteristic use. Detecting fraudulent accounts is another example.

With Amazon Neptune, you can use relationships to process financial and purchase transactions in near-real time to easily detect fraud patterns. Neptune provides a fully managed service to execute fast graph queries to detect that a potential purchaser is using the same email address and credit card as a known fraud case. If you are building a retail fraud detection application, Neptune can help you build
graph queries to easily detect relationship patterns like multiple people associated with a personal email address, or multiple people sharing the same IP address but residing in different physical addresses.

The following graph shows the relationship of three people and their identity-related information. Each person has an address, a bank account, and a social security number. However, we can see that Matt and Justin share the same social security number, which is irregular and indicates possible fraud by one or more of the connected people. A query to the graph database could help you discover these types of connections so that they can be reviewed.
Recommendation Engines

With Amazon Neptune, you can store relationships between information categories such as customer interests, friends, and purchase history in a graph. You can then quickly query it to make recommendations that are personalized and relevant. For example, you can use a highly available graph
database to make product recommendations to a user based on which products are purchased by others who follow the same sport and have similar purchase history. Or, you can identify people who have a friend in common, but don't yet know each other, and make a friendship recommendation.

Knowledge Graphs

Amazon Neptune helps you build knowledge graph applications. A knowledge graph lets you store information in a graph model and use graph queries to help your users navigate highly connected datasets more easily. Neptune supports open source and open standard APIs so that you can quickly use existing information resources to build your knowledge graphs and host them on a fully managed service. For example, if a user is interested in the Mona Lisa by Leonardo da Vinci, you can help them discover other works of art by the same artist or other works located in The Louvre. Using a knowledge graph, you can add topical information to product catalogs, build and query complex models of regulatory rules, or model general information, like Wikidata.

Life Sciences

Amazon Neptune helps you build applications that store and navigate information in the life sciences, and process sensitive data easily using encryption at rest. For example, you can use Neptune to store models of disease and gene interactions, and search for graph patterns within protein pathways to find other genes that may be associated with a disease. You can model chemical compounds as a graph and query for patterns in molecular structures. Neptune helps you integrate information to tackle challenges in healthcare and life sciences research. You can use Neptune to create and store patient relationships from medical records across different systems and topically organize research publications to find relevant information quickly.

Network / IT Operations

You can use Amazon Neptune to store a graph of your network and use graph queries to answer questions like how many hosts are running a specific application. Neptune can store and process billions of events to manage and secure your network. If you detect an event, you can use Neptune to quickly understand how it might affect your network by querying for a graph pattern using the attributes of the event. You can issue graph queries to Neptune to find other hosts or devices that may be compromised. For example, if you detect a malicious file on a host, Neptune can help you find the connections between the hosts that spread the malicious file and enable you to trace it to the original host that downloaded it.

Graph Queries and Traversals

Neptune supports two different graph query languages: Gremlin (Apache TinkerPop3) and SPARQL (SPARQL 1.1).

- Gremlin is a graph traversal language and, as such, a query in Gremlin is a traversal made up of discrete steps. Each step follows an edge to a node.
- SPARQL is a declarative query language based on graph pattern-matching standardized by the W3C.

Given the following graph of people (nodes) and their relationships (edges), you can find out who the “friends of friends” of a particular person are—for example, the friends of Howard's friends.
Looking at the graph, you can see that Howard has one friend, Jack, and Jack has three friends: Annie, Harry, and Mac. This is a simple example with a simple graph, but these types of queries can scale in complexity, dataset size, and result size.

The following is a Gremlin traversal query that returns the names of the friends of Howard's friends.

```
g.V().has('name', 'Howard').out('friend').out('friend').values('name')
```
The following is a SPARQL query that returns the names of the friends of Howard's friends.

Note
Each part of any Resource Description Framework (RDF) triple has a URI associated with it. In this example, the URI prefix is intentionally short. For more information, see Accessing the Neptune Graph with SPARQL (p. 44).

```
prefix : <#>

select ?names where {
  ?howard :name "Howard" .
}
```

For more examples of Gremlin and SPARQL queries, see Accessing a Neptune Graph (p. 24).
Amazon Neptune Quick Start Using AWS CloudFormation

This section contains steps and other information to help you get started quickly with Amazon Neptune. For general information about Neptune, see What Is Amazon Neptune? (p. 1).

These instructions use an AWS CloudFormation template to create the required resources for you. For instructions on creating these resources yourself, see Getting Started with Neptune (p. 17).

**Important**
The AWS CloudFormation stack that is created by this template creates multiple resources, including resources in Neptune, Amazon Elastic Compute Cloud (Amazon EC2), Amazon Virtual Private Cloud (Amazon VPC), and AWS Identity and Access Management (IAM). Some of these resources are not free-tier resources. For pricing information, see Amazon Neptune Pricing and Amazon EC2 Pricing. You can delete the stack when you are finished with it to stop any charges.

This AWS CloudFormation stack is intended as a basis for a tutorial for Amazon Neptune. We recommend you to use stricter IAM policies and security for your production environment if you use this template. For information a securing resources, see Amazon VPC Security and Amazon EC2 Network and Security.

**Topics**
- Prerequisites (p. 10)
- Launch the Amazon Neptune CloudFormation Stack (p. 11)
- Accessing the Neptune Graph (p. 12)

**Prerequisites**

Before you create an Amazon Neptune cluster, you need to have the following:

- The required IAM permissions.
- A key pair

**IAM Permissions**
The following permissions allow you to create resources for the AWS CloudFormation stack:

**AWS Managed Policies**

- AWSCloudFormationReadOnlyAccess
- NeptuneConsoleFullAccess
- AmazonEC2FullAccess

**Additional IAM Permissions**
The following policy outlines the additional permissions that are required to create and delete this CloudFormation Stack.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowEC2FullAccess",
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "ec2:CreateNetworkAcl",
                "ec2:CreateNetworkAclEntry",
                "ec2:CreateSubnet",
                "ec2:CreateRoute",
                "ec2:CreateVolume",
                "ec2:CreateSnapshot",
                "ec2:DeleteVolume",
                "ec2:DeleteSnapshot",
                "ec2:ModifyVolumeAttribute",
                "ec2:CreateKeyPair",
                "ec2:DeleteKeyPair",
                "ec2:Describe*",
                "ec2:RunInstances",
                "ec2:StopInstances",]
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalId": ["*"]
                }
            }
        }
```
"Statement": [{
  "Effect": "Allow",
  "Action": [
    "iam:GetSSHPublicKey",
    "iam:ListSSHPublicKeys",
    "iam:CreateRole",
    "iam:CreatePolicy",
    "iam:GetRolePolicy",
    "iam:CreateInstanceProfile",
    "iam:AddRoleToInstanceProfile",
    "iam:GetAccountSummary",
    "iam:ListAccountAliases",
    "iam:PassRole",
    "iam:GetRole",
    "cloudformation:*Stack",
    "iam:DeleteRole",
    "iam:RemoveRoleFromInstanceProfile",
    "iam:DeleteRolePolicy",
    "iam:DeleteInstanceProfile"
  ],
  "Resource": "*"
}]

Note
The following permissions are only required to delete a stack: "iam:DeleteRole",
"iam:RemoveRoleFromInstanceProfile", "iam:DeleteRolePolicy",
"iam:DeleteInstanceProfile.

EC2 Key Pair
You must have a key pair (and the PEM file) available in the Region that you create the AWS
CloudFormation stack in. If you need to create a key pair, see Creating a Key Pair Using Amazon EC2 for
instructions on creating the pair and downloading the PEM file.

Launch the Amazon Neptune CloudFormation Stack

1. To launch the Neptune stack in the AWS CloudFormation console, choose one of the Launch Stack
   buttons in the following table.

<table>
<thead>
<tr>
<th>Region</th>
<th>View</th>
<th>View in Designer</th>
<th>Launch</th>
</tr>
</thead>
<tbody>
<tr>
<td>US East (N. Virginia)</td>
<td>View</td>
<td>View in Designer</td>
<td>[Launch Stack]</td>
</tr>
<tr>
<td>US East (Ohio)</td>
<td>View</td>
<td>View in Designer</td>
<td>[Launch Stack]</td>
</tr>
<tr>
<td>US West (Oregon)</td>
<td>View</td>
<td>View in Designer</td>
<td>[Launch Stack]</td>
</tr>
<tr>
<td>EU (Ireland)</td>
<td>View</td>
<td>View in Designer</td>
<td>[Launch Stack]</td>
</tr>
</tbody>
</table>

2. On the Select Template page, choose Next.
3. On the Specify Details page, choose a key pair for the EC2SSHKeyPairName.
Accessing the Neptune Graph

Now that you have an instance, you can log into your EC2 instance using SSH and connect to the Neptune graph. For information about connecting to an EC2 instance using SSH, see Connect to Your Linux Instance in the Amazon EC2 User Guide for Linux Instances.

If you are using a Linux or macOS command line to connect to the EC2 instance, you can paste the SSH command from the SSHAccess item in the Outputs section of the AWS CloudFormation stack. This requires that you have the PEM file in the current directory and the PEM file permissions must be set to 400 (chmod 400 keypair.pem).

After you are connected, see the following sections, which contain information about using the Gremlin and SPARQL endpoints of Neptune.

Topics
- Gremlin (p. 12)
- RDF / SPARQL (p. 15)

Gremlin

The Gremlin Console allows you to experiment with TinkerPop graphs and queries in a REPL (read-eval-print loop) environment.

The following tutorial walks you through using Gremlin with Amazon Neptune, including how to add vertices, edges, properties, and more, while highlighting Neptune-specific Gremlin implementation differences.

To access Neptune using the Gremlin Console

1. Change directories into the unzipped folder.
   
   ```
   cd apache-tinkerpop-gremlin-console-3.3.2
   ```

2. Type the following command to run the Gremlin Console.
   
   ```
   bin/gremlin.sh
   ```

   You should see the following output:
   
   ```
   \,.../
   (o o)
   -----oooo-(3)-oooo-----
   plugin activated: tinkerpop.server
   plugin activated: tinkerpop.utilities
   ```
You are now at the `gremlin>` prompt. You will type the remaining steps at this prompt.

3. At the `gremlin>` prompt, type the following to connect to the Neptune DB instance.

```bash
:remote connect tinkerpop.server conf/neptune-remote.yaml
```

4. At the `gremlin>` prompt, type the following to switch to remote mode. This sends all Gremlin queries to the remote connection.

```bash
:remote console
```

5. **Add vertex with label and property**

```java
g.addV('person').property('name', 'justin')
```

The vertex is assigned a string ID containing a GUID. All vertex IDs are strings in Neptune.

6. **Add a vertex with custom id**

```java
g.addV('person').property(id, '1').property('name', 'marko')
```

The `id` property is not quoted. It is a keyword for the ID of the vertex. The vertex ID here is a string with the number 1 in it.

Normal property names must be contained in quotation marks.

7. **Change property or add property if it doesn’t exist**

```java
g.V('1').property(single, 'name', 'marko')
```

Here you are changing the `name` property for the vertex from the previous step. This removes all existing values from the `name` property.

If you didn’t specify `single`, it instead appends the value to the `name` property if it hasn’t done so already.

8. **Add property, but append property if property already has a value.**

```java
g.V('1').property('age', 29)
```

Neptune uses set cardinality as the default action.

This command adds the `age` property with the value 29, but it does not replace any existing values.

If the `age` property already had a value, this command appends 29 to the property. For example, if the `age` property was 27, the new value would be `[27, 29]`.

9. **Add multiple vertices:**

```java
g.addV('person').property(id, '2').property('name', 'vadas').property('age', 27).next()
g.addV('software').property(id, '3').property('name', 'lop').property('lang', 'java').next()
g.addV('person').property(id, '4').property('name', 'josh').property('age', 32).next()
g.addV('software').property(id, '5').property('name', 'ripple').property('ripple', 'java').next()
g.addV('person').property(id, '6').property('name', 'peter').property('age', 35)
```
You can send multiple statements at the same time to Neptune. Statements can be separated by newline (``\n``), spaces (`` ` `), semicolon (`` ; ``), or nothing (for example: `g.addV('person').next().g.V()` is valid).

**Note**
The Gremlin Console sends a separate command at every newline (``\n``), so they will each be a separate transaction in that case. This example has all the commands on separate lines for readability. Remove the newline (``\n``) characters to send it as a single command via the Gremlin Console.

All statements other than the last statement must end in a terminating step, such as `.next()` or `.iterate()`, or they will not run. The Gremlin Console does not require these terminating steps. All statements that are sent together are included in a single transaction and succeed or fail together.

10. **Add edges**

```
g.V('1').addE('knows').to(g.V('2')).property('weight', 0.5).next()
g.addE('knows').from(g.V('1')).to(g.V('4')).property('weight', 1.0)
```

Here are two different ways to add an edge.

11. **Add the rest of the Modern graph**

```
g.V('1').addE('created').to(g.V('3')).property('weight', 0.4).next()
g.V('4').addE('created').to(g.V('5')).property('weight', 1.0).next()
g.V('4').addE('knows').to(g.V('3')).property('weight', 0.4).next()
g.V('6').addE('created').to(g.V('3')).property('weight', 0.2)
```

12. **Delete a Vertex**

```
g.V().has('name', 'justin').drop()
```

Removes the vertex with the `name` property equal to `justin`.

**Important**
*Stop here, and you have the full Apache TinkerPop Modern graph. The examples in the Traversal section of the TinkerPop documentation use the Modern graph.*

13. **Run a Traversal**

```
g.V().hasLabel('person')
```

Returns all `person` vertices.

14. **Run a Traversal with values (valueMap())**

```
g.V().has('name', 'marko').out('knows').valueMap()
```

Returns key, value pairs for all vertices that `marko` "knows."

15. **Multiple labels**

```
g.addV("Label1::Label2::Label3")
```

Neptune supports multiple labels for a vertex. When you create a label, you can specify multiple labels by separating them with `::`.
This example adds a vertex with three different labels.

The hasLabel step matches this vertex with any of those three labels: hasLabel("Label1"), hasLabel("Label2"), and hasLabel("Label3").

The :: delimiter is reserved for this use only.

You cannot specify multiple labels in the hasLabel step. For example, hasLabel("Label1::Label2") does not match anything.

16. **Time / date**

```java
g.V().property(single, 'lastUpdate', datetime('2018-01-01T00:00:00'))
```

Neptune does not support Java Date. Use the datetime() function instead. datetime() accepts an ISO8061-compliant datetime string.

It supports the following formats: YYYY-MM-DD, YYYY-MM-DDTHH:mm, YYYY-MM-DDTHH:mm:SS, YYYY-MM-DDTHH:mm:SSZ

17. **Delete vertices, properties, or edges**

```java
g.V().hasLabel('person').properties('age').drop().iterate()
g.V('1').drop().iterate()
g.V().outE().hasLabel('created').drop()
```

Here are several drop examples.

**Note**
The .next() step does not work with .drop(). Use .iterate() instead.

18. When you are finished, type the following to exit the Gremlin Console.

```java
:exit
```

**Note**
Use a semicolon (;) or a newline character (\n) to separate each statement.
Each traversal preceding the final traversal must end in next() to be executed. Only the data from the final traversal is returned.

For more information on the Neptune implementation of Gremlin, see the section called “Neptune Gremlin Implementation Differences” (p. 27).

**Additional Links**
- Accessing the Neptune Graph with Gremlin (p. 26)
- Loading Data into Neptune (p. 59)
- What Is Amazon Neptune? (p. 1)
- Getting Started with Neptune (p. 17)

**RDF / SPARQL**

SPARQL is a query language for the Resource Description Framework (RDF), which is a graph data format designed for the web. Amazon Neptune is compatible with SPARQL 1.1. This means that you can connect to a Neptune DB instance and query the graph using the query language described in the SPARQL 1.1 Query Language specification.
A query in SPARQL consists of a SELECT clause to specify the variables to return and a WHERE clause to specify which data to match in the graph. If you are unfamiliar with SPARQL queries, see Writing Simple Queries in the SPARQL 1.1 Query Language.

The HTTP endpoint for SPARQL queries to a Neptune DB instance is http://<your-neptune-endpoint>:8182/sparql.

To connect to SPARQL

1. You can get the SPARQL endpoint for your Neptune cluster from the SparqlEndpoint item in the Outputs section of the AWS CloudFormation stack.
2. Type the following to submit a SPARQL UPDATE using HTTP POST and the curl command.


   The preceding example inserts the following triple into the SPARQL default graph: <http://test.com/s> <http://test.com/p> <http://test.com/o>

3. Type the following to submit a SPARQL QUERY using HTTP POST and the curl command.


   The preceding example returns up to 10 of the triples (subject-predicate-object) in the graph by using the ?s ?p ?o query with a limit of 10. To query for something else, replace it with another SPARQL query.

   Note
   The default MIME type of a response is application/sparql-results+json for SELECT and ASK queries.
   The default MIME type of a response is application/n-quads for CONSTRUCT and DESCRIBE queries.
   For a list of all available MIME types, see SPARQL HTTP API (p. 51).

For more information about the Neptune SPARQL REST interface, see SPARQL HTTP API (p. 51). For more information about Amazon Neptune, see Next Steps (p. 55).

Additional Links

- Accessing the Neptune Graph with SPARQL (p. 44)
- Loading Data into Neptune (p. 59)
- What Is Amazon Neptune? (p. 1)
- Getting Started with Neptune (p. 17)
Getting Started with Neptune

This section provides details on the requirements and prerequisites for Amazon Neptune and shows you how to use it to create a Neptune DB instance. For a less detailed overview, see the Amazon Neptune Quick Start Using AWS CloudFormation (p. 10).

Topics
- Setting Up Amazon Neptune (p. 17)
- Launching a Neptune DB Cluster (p. 20)
- Creating a Neptune Replica Using the Console (p. 22)

Setting Up Amazon Neptune

Before you create a Neptune DB instance, you must have an Amazon Virtual Private Cloud (VPC). If you want to access your Neptune DB instance from outside the VPC, you must also have a security group for the VPC with rules that allow you to connect to the Neptune DB instance.

You also need an IAM user with NeptuneFullAccess permissions. This is required to use the Neptune console and create a Neptune cluster. For information about adding these permissions, see AWS Managed (Predefined) Policies for Amazon Neptune (p. 138).

The NeptuneFullAccess IAM policy does not grant permissions for VPC, VPC endpoints, or S3. You must add these additional permissions to your IAM user or role to operate on other services. For example, if you choose to have the Neptune Console create a VPC, the IAM user or role must have VPC permissions.

Note
Neptune requires permission to create a service-linked role the first time you create any Neptune resources. Add the following iam:CreateServiceLinkedRole permissions to the same user or role that you give NeptuneFullAccess.

```json
{
   "Action": "iam:CreateServiceLinkedRole",
   "Effect": "Allow",
   "Resource": "arn:aws:iam::*:role/aws-service-role/rds.amazonaws.com/AWSServiceRoleForRDS",
   "Condition": {
      "StringLike": {
         "iam:AWSServiceName": "rds.amazonaws.com"
      }
   }
}
```

For more information, see Using Service-Linked Roles for Neptune (p. 139).

Neptune VPC Requirements

If you created your AWS account after 2013–12–04, then you have a default VPC in each AWS Region. If you aren’t sure whether you have a default VPC, see the Detecting Whether You Have a Default VPC section in the Amazon VPC User Guide.

For more information about the default VPC, see Default VPC and Default Subnets in the Amazon VPC User Guide.
If you have a default VPC, you can create a VPC security group to allow an Amazon EC2 instance to connect to the Neptune DB instance from within the VPC. Access from the internet is allowed only to the EC2 instance. The EC2 instance is allowed access to the graph database.

There are many possible ways to configure a VPC or multiple VPCS. For information about creating your own VPCs, see the *Amazon VPC User Guide*.

An Amazon Neptune DB cluster can only be created in an Amazon VPC that has at least two subnets in at least two Availability Zones. By distributing your cluster instances across at least two Availability Zones, you help ensure that there are instances available in your DB cluster in the unlikely event of an Availability Zone failure. The cluster volume for your Neptune DB cluster always spans three Availability Zones to provide durable storage with less possibility of data loss.

If you're using the Amazon Neptune console to create your Neptune DB cluster, you can have Neptune automatically create a VPC for you. Alternatively, you can use an existing VPC or create a new VPC for your Neptune DB cluster. Your VPC must have at least two subnets in order for you to use it with an Amazon Neptune DB cluster.

**Note**
You can communicate with an Amazon EC2 instance that is not in a VPC and a Neptune DB cluster using ClassicLink.

If you don't have a default VPC, and you have not created a VPC, you can have Neptune automatically create a VPC for you when you create a Neptune DB cluster using the console. Neptune can also create a VPC security group and a DB subnet group for you.

Otherwise, you must do the following:

- Create a VPC with at least two subnets in at least two Availability Zones.
• Specify a VPC security group that authorizes connections to your Neptune DB cluster. You can do this in the Amazon VPC console at https://console.aws.amazon.com/vpc/.
• Specify a Neptune DB subnet group with at least two subnets with each subnet in a different Availability Zone. You can create a DB subnet group in the Neptune console at https://console.aws.amazon.com/neptune/home.

You must create a Neptune DB subnet group using the console. Amazon RDS DB subnet groups don’t work with Neptune.

Note
Amazon Neptune is not supported in every Availability Zone. If you receive the console error DB Subnet Group doesn’t meet availability zone coverage requirement, try adding subnets in additional Availability Zones to the DB subnet group.

The following section walks you through setting up a security group for your default VPC, as shown in the preceding diagram.

Creating a Security Group to Provide Access to the Neptune DB Instance in the VPC

Your Neptune DB instance is launched in a VPC. Security groups provide access to the Neptune DB instance in the VPC. They act as a firewall for the associated Neptune DB instance, controlling both inbound and outbound traffic at the instance level. Neptune DB instances are created by default with a firewall and a default security group that prevents access to the Neptune DB instance. You must add rules to a security group that enable you to connect to your DB instance.

The security group that you need to create is a VPC security group. Neptune DB instances in a VPC require that you add rules to a VPC security group to allow access to the instance.

The following procedure shows you how to add a custom TCP rule that specifies the port range and IP addresses that the EC2 instance uses to access the database. You can use the VPC security group assigned to the EC2 instance rather than the IP address.

To create a VPC security group for Neptune

1. Sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the upper-right corner of the console, choose the AWS Region in which you want to create the VPC security group and the Neptune DB instance. In the list of Amazon VPC resources for that Region, it should show that you have at least one VPC and several subnets. If it does not, you don’t have a default VPC in that Region.
3. In the navigation pane, choose Security Groups.
5. In the Create Security Group window, type the Name tag, Group name, and Description of your security group. Choose the VPC that you want to create your Neptune DB instance in. Choose Yes, Create.
6. The VPC security group that you created should still be selected. The details pane at the bottom of the console window displays the details for the security group, and tabs for working with inbound and outbound rules. Choose the Inbound Rules tab.
7. On the Inbound Rules tab, choose Edit. In the Type list, choose Custom TCP Rule.
8. In the PortRange text box, type 8182, the default port value for a Neptune DB instance. Then type the IP address range (CIDR value) from where you will be accessing the instance, or choose a security group name in the Source text box.
9. If you need to add more IP addresses or different port ranges, choose **Add another rule**.
10. When you have finished, choose **Save**.

You will use the VPC security group you just created as the security group for your DB instance when you create it.

Finally, a quick note about VPC subnets: If you use a default VPC, a default subnet group spanning all of the VPC's subnets is already created for you. When you use the **Launch a Neptune DB instance** wizard to create a DB instance, you can choose the default VPC and use default for the **DB Subnet Group**.

After you complete the setup requirements, you can use your settings and the security group you created to launch a Neptune DB instance.

## Launching a Neptune DB Cluster

The following procedures describe how to use the AWS Management Console to launch an Amazon Neptune DB cluster and create a Neptune Replica.

### Launch a Neptune DB Cluster Using the Console

Before you can access the Neptune console and create a Neptune cluster, you must have an IAM user with **NeptuneFullAccess** permissions. For information about adding these permissions, see AWS Managed (Predefined) Policies for Amazon Neptune (p. 138).

The **NeptuneFullAccess** IAM policy does not grant permissions for VPC, VPC endpoints, or S3. You must add these additional permissions to your IAM user or role to operate on other services. For example, if you choose to have the Neptune Console create a VPC, the IAM user or role must have VPC permissions.

**Note**

Neptune requires permission to create a service-linked role the first time you create any Neptune resources. Add the following **iam:CreateServiceLinkedRole** permissions to the same user or role that you give **NeptuneFullAccess**.

```json
{
  "Action": "iam:CreateServiceLinkedRole",
  "Effect": "Allow",
  "Resource": "arn:aws:iam::*:role/aws-service-role/rds.amazonaws.com/AWSServiceRoleForRDS",
  "Condition": {
    "StringLike": {
      "iam:AWSServiceName": "rds.amazonaws.com"
    }
  }
}
```

For more information, see Using Service-Linked Roles for Neptune (p. 139).

### To launch a Neptune DB cluster using the console

2. Choose **Launch DB Instance** to start the **Launch DB instance** wizard.
3. On the **Specify DB details** page, you can customize the settings for your Neptune DB cluster. The following table shows the advanced settings for a DB cluster.
Launch a Neptune DB Cluster Using the Console

For this option... | Do this
---|---
**DB instance class** | Choose a DB instance class that defines the processing and memory requirements for each instance in the DB cluster. Neptune supports the db.r4.large, db.r4.xlarge, db.r4.2xlarge, db.r4.4xlarge, db.r4.8xlarge DB instance classes.

**DB instance identifier** | Type a name for the primary instance in your DB cluster. This identifier is used in the endpoint address for the primary instance of your DB cluster.

The DB instance identifier has the following constraints:

- It must contain from 1 to 63 alphanumeric characters or hyphens.
- Its first character must be a letter.
- It cannot end with a hyphen or contain two consecutive hyphens.
- It must be unique for all DB instances per AWS account, per AWS Region.

4. On the **Configure advanced settings** page, you can customize additional settings for your Neptune DB cluster. The following table shows the advanced settings for a DB cluster.

<table>
<thead>
<tr>
<th>For this option...</th>
<th>Do this</th>
</tr>
</thead>
</table>
**Virtual Private Cloud (VPC)** | Choose the VPC that will host the DB cluster. Choose **Create a new VPC** to have Neptune create a VPC for you. You need to create an Amazon EC2 instance in this same VPC to access the Neptune instance. For more information, see Setting Up Amazon Neptune (p. 17).

**Subnet group** | Choose the Neptune DB subnet group to use for the DB cluster. If your VPC does not have any subnet groups, Neptune creates a DB subnet group for you. For more information, see Setting Up Amazon Neptune (p. 17).

**Availability Zone** | Specify a particular Availability Zone, or choose **No preference** to have Neptune choose one for you.

**VPC security groups** | Choose one or more VPC security groups to secure network access to the DB cluster. Choose **Create a new VPC security group** to have Neptune create a VPC security group for you. For more information, see Setting Up Amazon Neptune (p. 17).

**DB Cluster Identifier** | The identifier for your DB cluster. If you don’t specify this value, Neptune creates one based on the DB instance identifier.

**Database port** | The port for all HTTP and WebSockets connections. Neptune DB clusters use 8182 as the default.
<table>
<thead>
<tr>
<th>For this option...</th>
<th>Do this</th>
</tr>
</thead>
</table>
| IAM DB Authentication | Choose **Enable IAM DB authentication** to manage access and authentication using AWS Identity and Access Management.  
**Important**  
This requires that you sign all requests with AWS Signature version 4 signing. For more information, see [IAM Database Authentication for Neptune](p. 122) |
| Enable encryption | Choose **Yes** to enable encryption at rest for this DB cluster. For more information, see [Encrypting Neptune Resources](p. 104). |
| Failover priority | Choose the priority tier. If there is contention within a tier, the replica that is the same size as the primary instance is selected. |
| Backup retention period | Choose the length of time, from 1 to 35 days, that Neptune will retain backup copies of the database. Backup copies can be used for point-in-time restores (PITR) of your database down to the second. |
| Auto minor version upgrade | Choose **Yes** if you want to enable your Neptune DB cluster to receive minor Neptune DB Engine version upgrades automatically when they become available.  
The **Auto minor version upgrade** option applies only to upgrades to Neptune minor engine versions for your Amazon Neptune DB cluster. It doesn't apply to regular patches that are applied to maintain system stability. |
| Maintenance window | Choose the weekly time range during which system maintenance can occur. |

5. Choose **Launch DB instance** to launch your Neptune DB instance, and then choose **Close** to close the wizard.

On the Amazon Neptune console, the new DB cluster appears in the list of DB clusters. The DB cluster has a status of **creating** until it is created and ready for use. When the state changes to **available**, you can connect to the primary instance for your DB cluster. Depending on the DB instance class and store allocated, it can take several minutes for the new instances to be available.

To view the newly created cluster, choose the **Clusters** view in the Neptune console.

Note the **Cluster endpoint** value. You will need this to connect to your Neptune DB cluster.

---

**Creating a Neptune Replica Using the Console**

After creating the primary instance for your Neptune DB cluster, you can add additional Neptune Replicas by using the **Create read replica** page.

**To create a Neptune Replica by using the AWS Management Console.**

2. In the left navigation pane, choose **Instances**.
3. Choose to select the check box to the left of the primary instance for your Neptune DB cluster.
4. Choose **Instance Actions**, and then choose **Create read replica**.
5. On the **Create replica DB instance** page, specify options for your Neptune Replica. The following table shows settings for a Neptune read replica.

<table>
<thead>
<tr>
<th>For This Option...</th>
<th>Do This</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Availability zone</strong></td>
<td>Specify an Availability Zone. Choose a different zone than the primary DB instance. The list includes only those Availability Zones that are mapped by the DB subnet group for the DB cluster.</td>
</tr>
<tr>
<td><strong>Encryption</strong></td>
<td>Enable or disable encryption.</td>
</tr>
<tr>
<td><strong>DB instance class</strong></td>
<td>Select a DB instance class that defines the processing and memory requirements for the Neptune Replica. Neptune supports the <code>db.r4.large</code>, <code>db.r4.xlarge</code>, <code>db.r4.2xlarge</code>, <code>db.r4.4xlarge</code>, <code>db.r4.8xlarge</code> DB instance classes.</td>
</tr>
<tr>
<td><strong>Read replica source</strong></td>
<td>Select the identifier of the primary instance to create an Neptune Replica for.</td>
</tr>
<tr>
<td><strong>DB instance identifier</strong></td>
<td>Type a name for the instance that is unique for your account in the region you selected. You might choose to add some intelligence to the name such as including the availability zone selected, for example <code>neptune-us-east-1c</code>.</td>
</tr>
<tr>
<td><strong>DB parameter group</strong></td>
<td>The parameter group for this instance.</td>
</tr>
<tr>
<td><strong>Auto Minor Version Upgrade</strong></td>
<td>Select <strong>Yes</strong> if you want to enable your Neptune Replica to receive minor Neptune DB engine version upgrades automatically when they become available. The <strong>Auto Minor Version Upgrade</strong> option only applies to upgrades to MySQL minor engine versions for your Amazon Neptune; DB cluster. It doesn't apply to regular patches applied to maintain system stability.</td>
</tr>
</tbody>
</table>

6. Choose **Create read replica** to create the Neptune replica instance.
Accessing a Neptune Graph

Amazon Neptune supports two different graph query languages: Gremlin (Apache TinkerPop3) and SPARQL (SPARQL 1.1). Instructions for accessing the Neptune graph on a running Neptune DB instance are divided into sections for Gremlin and SPARQL.

Gremlin data and SPARQL data are separated, and any data that is loaded or stored with one query language cannot be queried by the other.

Gremlin

Gremlin is a graph traversal language, and as such, a query in Gremlin is a traversal made up of discrete steps. Each step follows an edge to a node.

To learn about connecting to Neptune with Gremlin, see Accessing the Neptune Graph with Gremlin (p. 26).

SPARQL

SPARQL is a declarative query language based on the graph pattern matching that is standardized by the W3C and described in the SPARQL 1.1 Query Language specification.

To learn about connecting to Neptune with SPARQL, see Accessing the Neptune Graph with SPARQL (p. 44).

Topics

• Finding the Endpoint for a Neptune Cluster (p. 24)
• Launching an Amazon EC2 Instance (p. 25)
• Accessing the Neptune Graph with Gremlin (p. 26)
• Accessing the Neptune Graph with SPARQL (p. 44)
• Check the Health Status of a Neptune Cluster (p. 55)
• Neptune Service Errors (p. 55)

Finding the Endpoint for a Neptune Cluster

To run the examples in this guide, you must have the endpoint for an Amazon Neptune cluster. The following procedure shows you how to get this information.

To find the endpoint for a Neptune cluster

1. Sign in to the AWS Management Console, and open the Amazon Neptune console at https://console.aws.amazon.com/neptune/home.
2. Choose Clusters, and then choose the DB cluster from the list.
3. Choose the Details tab to show the DB cluster details. On the Details page, copy the value for the Cluster endpoint.
Access to Neptune is limited to within the virtual private cloud (VPC) that the Neptune DB instance is in. The following steps launch an Amazon Elastic Compute Cloud (Amazon EC2) instance in your default Amazon VPC. If you created a Neptune DB instance in a different VPC, you should launch the EC2 instance in that VPC.

**To launch an EC2 instance**

1. Open the Amazon EC2 console at [https://console.aws.amazon.com/ec2/](https://console.aws.amazon.com/ec2/).
2. In the upper-right corner of the console window, choose **US East (N. Virginia)** from the Region selector.
3. Choose **Launch Instance**, and do the following:
   a. Choose an Amazon Machine Image (AMI):
      - At the top of the list of AMIs, go to **Amazon Linux AMI**, and choose **Select**.
   b. Choose an Instance Type:
      1. At the top of the list of instance types, choose **t2.micro**.
      2. Choose **Next: Configure Instance Details**.
   c. Configure Instance Details:
1. Go to Network, and choose your default VPC.
2. Choose Next: Add Storage.

d. Add Storage:
   - Skip this step by choosing Next: Tag Instance.

e. Tag Instance:
   - Skip this step by choosing Next: Configure Security Group.

f. Configure Security Group:
   1. Choose Select an existing security group.
   2. In the list of security groups, choose default. This is the default security group for your VPC.

g. Review Instance Launch:
   - Choose Launch.

4. In the Select an existing key pair or create a new key pair window, do one of the following:
   - If you don't have an Amazon EC2 key pair, choose Create a new key pair and follow the instructions. You are asked to download a private key file (.pem file); you need this file when you log in to your Amazon EC2 instance.
   - If you already have an existing Amazon EC2 key pair, go to Select a key pair and choose your key pair from the list. You must already have the private key file (.pem file) available in order to log in to your Amazon EC2 instance.

5. When you have configured your key pair, choose Launch Instances.

6. Return to the Amazon EC2 console home page and choose the instance that you launched. In the lower pane, on the Description tab, find the Public DNS for your instance; for example: ec2-00-00-00-00.us-east-1.compute.amazonaws.com.

   Make a note of this public DNS name, because you need it to connect to the instance.

   **Note**
   It takes a few minutes for your Amazon EC2 instance to become available. Before you continue, ensure that the Instance State is running and that all of its Status Checks have passed.

---

### Accessing the Neptune Graph with Gremlin

Amazon Neptune is compatible with Apache TinkerPop3 and Gremlin 3.3.2. This means that you can connect to a Neptune DB instance and use the Gremlin traversal language to query the graph.

A traversal in Gremlin is a series of chained steps. It starts at a vertex (or edge) and walks the graph by following the outgoing edges of each vertex and then the outgoing edges of those vertices. Each step is an operation in the traversal. For more information, see The Traversal in the TinkerPop3 documentation.

There are Gremlin language variants and support for Gremlin access in various programming languages. For more information, see On Gremlin Language Variants in the TinkerPop3 documentation.

This documentation describes how to access Neptune with the following variants and programming languages.

**Gremlin-Groovy**

The Gremlin Console and HTTP REST examples in this section use the Gremlin-Groovy variant.
Neptune Gremlin Implementation Differences

There are a few important differences between the Amazon Neptune implementation of Gremlin and the TinkerPop implementation.

Pre-Bound Variables

The traversal object `g` is Pre-bound. The `graph` object is not supported.

TinkerPop Enumerations

Neptune does not support fully qualified class names for enumeration values. For example, you must use `single` and not `org.apache.tinkerpop.gremlin.structure.VertexProperty.Cardinality.single` in your Groovy request.

The enumeration type is determined by parameter type.

The following table shows the allowed enumeration values and the related TinkerPop fully qualified name.
### Allowed Values

<table>
<thead>
<tr>
<th>id, key, label, value</th>
<th>org.apache.tinkerpop.gremlin.structure.T</th>
</tr>
</thead>
<tbody>
<tr>
<td>T.id, T.key, T.label, T.value</td>
<td>org.apache.tinkerpop.gremlin.structure.T</td>
</tr>
<tr>
<td>set, single</td>
<td>org.apache.tinkerpop.gremlin.structure.VertexProperty.Cardinality</td>
</tr>
<tr>
<td>decr, incr, shuffle</td>
<td>org.apache.tinkerpop.gremlin.process.traversal.Order</td>
</tr>
<tr>
<td>Order.decr, Order.incr, Order.shuffle</td>
<td>org.apache.tinkerpop.gremlin.process.traversal.Order</td>
</tr>
<tr>
<td>global, local</td>
<td>org.apache.tinkerpop.gremlin.process.traversal.Scope</td>
</tr>
<tr>
<td>Scope.global, Scope.local</td>
<td>org.apache.tinkerpop.gremlin.process.traversal.Scope</td>
</tr>
<tr>
<td>all, first, last, mixed</td>
<td>org.apache.tinkerpop.gremlin.process.traversal.Pop</td>
</tr>
<tr>
<td>normSack</td>
<td>org.apache.tinkerpop.gremlin.process.traversal.SackFunctions.Barrier</td>
</tr>
<tr>
<td>addAll, and, assign, div, max, min, minus, mult, or, sum, sumLong</td>
<td>org.apache.tinkerpop.gremlin.process.traversal.Operator</td>
</tr>
<tr>
<td>keys, values</td>
<td>org.apache.tinkerpop.gremlin.structure.Column</td>
</tr>
<tr>
<td>BOTH, IN, OUT</td>
<td>org.apache.tinkerpop.gremlin.structure.Direction</td>
</tr>
</tbody>
</table>

### Java Code

Neptune does not support calls to methods defined by arbitrary Java or Java library calls other than supported Gremlin APIs. For example, `java.lang.*`, `Date()`, and `g.V().tryNext().orElseGet()` are not allowed.

### Date and Time

Neptune does not support the Java `Date()` function. A date and time value must be specified with the `datetime()` function.

The `datetime()` function takes a string value with an ISO8061 compliant datetime. For example, `datetime('2018-01-01T00:00:00')`. Supports the following formats: `YYYY-MM-DD`, `YYYY-MM-DDTHH:mm`, `YYYY-MM-DDTHH:mm:SS`, `YYYY-MM-DDTHH:mm:SSZ`

### Script Execution

All queries must begin with `g`, the traversal object.

Multiple traversals can be issued separated by a semicolon (;) or a newline character (\n). Every statement other than the last must end with a `next()` step to be executed. Only the final traversal data is returned.

### Sessions

Neptune is sessionless. It does not support the console `session` argument. For a description of the difference, see the TinkerPop Session Reference.

### Transactions

Neptune opens a new transaction at the beginning of each Gremlin traversal and closes the transaction upon the successful completion of the traversal. The transaction is rolled back when there is an error.
Multiple statements separated by a semicolon (;) or a newline character (\n) are included in a single transaction. Every statement other than the last must end with a `next()` step to be executed. Only the final traversal data is returned.

Manual transaction logic using `tx.commit()` and `t.rollback()` is not supported.

**Vertex and Edge IDs**

Neptune Gremlin Vertex and Edge IDs must be of type `String`. If you don't supply an ID when you add a vertex or an edge, a UUID is generated and converted to a string; for example, “48af8178-50ce-971a-fc41-8c9a954cea62”.

- **Note**
  - This means that user-supplied IDs are supported, but they are optional in normal usage.
  - However, the Neptune `Load` command requires that all IDs be specified using the `~id` field in the Neptune CSV format.

**User Supplied IDs**

User supplied IDs are allowed in Neptune Gremlin with the following stipulations.

- Supplied IDs are optional.
- Only vertexes and edges are supported.
- Only type `String` is supported.

To create a new vertex with a custom ID, use the `property` step with the `id` keyword: `g.addV().property(id, 'customid')`.

- **Note**
  - Do not put quotation marks around the `id` keyword.

If you try to create a new vertex using the `g.addV()` and a vertex with that ID already exists, the operation fails. The exception to this is that if you specify a new label for the vertex, the operation succeeds. Any additional properties specified are added to the vertex. Nothing is overwritten. For example, the following Gremlin Console commands will succeed:

```javascript
gremlin> g.addV('label1').property(id, 'customid')
```

```javascript
gremlin> g.addV('label2').property(id, 'customid')
```

```javascript
gremlin> g.V('customid').label()
==>'label1::label2
```

**Vertex Property IDs**

Vertex property IDs are generated automatically and can show up as positive or negative numbers when queried.

**Cardinality of Vertex Properties**

Neptune supports set cardinality and single cardinality. Set cardinality is selected if not specified. This means that if you set a property value, it adds a new value to the property, but only if it does not already appear in the set of values. This is the Gremlin enumeration value of `Set`.

List is not supported. For more information about property cardinality, see the Vertex topic in the Gremlin JavaDoc.

**Updating a Vertex Property**

To update a property value without adding an additional value to the set of values, specify `single` cardinality in the `property` step.
Note
This removes all existing values for the property.

Labels
Neptune supports multiple labels for a vertex. When you create a label, you can specify multiple labels by separating them with ::. For example, `g.addV("Label1::Label2::Label3")` adds a vertex with three different labels. The `hasLabel` step matches this vertex with any of those three labels: `hasLabel("Label1")`, `hasLabel("Label2")`, and `hasLabel("Label3")`.

Important
The :: delimiter is reserved for this use only. You cannot specify multiple labels in the `hasLabel` step. For example, `hasLabel("Label1::Label2")` does not match anything.

Variables
Neptune does not support Gremlin variables and does not support the `bindings` property.

Timeouts
The Gremlin Console command `:remote config timeout` sets the local timeout only. To set the remote query timeout for Neptune, use the `neptune_query_timeout` parameter. For more information, see Amazon Neptune DB Parameter Groups (p. 94).

Groovy Limitations
Neptune doesn't support Groovy commands that don't start with g. This includes math (for example: `1+1`), system calls (for example: `System.nanoTime()`), and variable definitions (for example: `1+1`).

Important
Neptune does not support fully qualified class names. For example, you must use `single` and not `org.apache.tinkerpop.gremlin.structure.VertexProperty.Cardinality.single` in your Groovy request.

Serialization
Neptune supports the following serializations based on the requested MIME type.

<table>
<thead>
<tr>
<th>MIME type</th>
<th>Serialization</th>
</tr>
</thead>
<tbody>
<tr>
<td>application/vnd.gremlin-v1.0+gyro</td>
<td>GryoMessageSerializerV1d0</td>
</tr>
<tr>
<td>application/vnd.gremlin-v1.0+gyro-stringd</td>
<td>GryoMessageSerializerV1d0</td>
</tr>
<tr>
<td>application/vnd.gremlin-v3.0+gyro</td>
<td>GryoMessageSerializerV3d0</td>
</tr>
<tr>
<td>application/vnd.gremlin-v3.0+gyro-stringd</td>
<td>GryoMessageSerializerV3d0</td>
</tr>
<tr>
<td>application/vnd.gremlin-v1.0+json</td>
<td>GraphSONMessageSerializerGremlinV1d0</td>
</tr>
<tr>
<td>application/vnd.gremlin-v2.0+json</td>
<td>GraphSONMessageSerializerGremlinV2d0</td>
</tr>
<tr>
<td>application/json</td>
<td>GraphSONMessageSerializerV3d0</td>
</tr>
</tbody>
</table>

Lambda Steps
Neptune does not support Lambda Steps.
Unsupported Gremlin Methods

Neptune does not support the following Gremlin methods:

- `org.apache.tinkerpop.gremlin.process.traversal.dsl.graph.GraphTraversal.program(org.apache.tinkerpop.gremlin.process.computer.VertexProgram)`
- `org.apache.tinkerpop.gremlin.process.traversal.dsl.graph.GraphTraversal.sideEffect(java.util.function.Consumer)`
- `org.apache.tinkerpop.gremlin.process.traversal.dsl.graph.GraphTraversal.from(org.apache.tinkerpop.gremlin.structure.Vertex)`
- `org.apache.tinkerpop.gremlin.process.traversal.dsl.graph.GraphTraversal.to(org.apache.tinkerpop.gremlin.structure.Vertex)`

For example, the following traversal is not allowed: `g.V().from(g.V().next())`.

Other Features

The Neptune implementation of Gremlin does not expose the `graph` object. The following section describes the supported and unsupported `graph` features.

Gremlin Graph Supported Features

Here is a set of features as implemented by the Neptune Gremlin graph. These features are the same as would be returned by the `graph.features()` command.

<table>
<thead>
<tr>
<th>Graph Feature</th>
<th>Enabled</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transactions</td>
<td>true</td>
</tr>
<tr>
<td>ThreadedTransactions</td>
<td>false</td>
</tr>
<tr>
<td>Computer</td>
<td>false</td>
</tr>
<tr>
<td>Persistence</td>
<td>true</td>
</tr>
<tr>
<td>ConcurrentAccess</td>
<td>true</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable Feature</th>
<th>Enabled</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variables</td>
<td>false</td>
</tr>
<tr>
<td>SerializableValues</td>
<td>false</td>
</tr>
<tr>
<td>UniformListValues</td>
<td>false</td>
</tr>
<tr>
<td>BooleanArrayValues</td>
<td>false</td>
</tr>
<tr>
<td>DoubleArrayValues</td>
<td>false</td>
</tr>
<tr>
<td>IntegerArrayValues</td>
<td>false</td>
</tr>
<tr>
<td>StringArrayValues</td>
<td>false</td>
</tr>
<tr>
<td>BooleanValues</td>
<td>false</td>
</tr>
<tr>
<td>ByteValues</td>
<td>false</td>
</tr>
<tr>
<td>DoubleValues</td>
<td>false</td>
</tr>
<tr>
<td>FloatValues</td>
<td>false</td>
</tr>
<tr>
<td>IntegerValues</td>
<td>false</td>
</tr>
<tr>
<td>Feature</td>
<td>Value</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>LongValues</td>
<td>false</td>
</tr>
<tr>
<td>MapValues</td>
<td>false</td>
</tr>
<tr>
<td>MixedListValues</td>
<td>false</td>
</tr>
<tr>
<td>StringValues</td>
<td>false</td>
</tr>
<tr>
<td>ByteArrayValues</td>
<td>false</td>
</tr>
<tr>
<td>FloatArrayValues</td>
<td>false</td>
</tr>
<tr>
<td>LongArrayValues</td>
<td>false</td>
</tr>
<tr>
<td>Vertex Feature</td>
<td>Enabled</td>
</tr>
<tr>
<td>MetaProperties</td>
<td>false</td>
</tr>
<tr>
<td>DuplicateMultiProperties</td>
<td>false</td>
</tr>
<tr>
<td>AddVertices</td>
<td>true</td>
</tr>
<tr>
<td>RemoveVertices</td>
<td>true</td>
</tr>
<tr>
<td>MultiProperties</td>
<td>true</td>
</tr>
<tr>
<td>UserSuppliedIds</td>
<td>true</td>
</tr>
<tr>
<td>AddProperty</td>
<td>true</td>
</tr>
<tr>
<td>RemoveProperty</td>
<td>true</td>
</tr>
<tr>
<td>NumericIds</td>
<td>false</td>
</tr>
<tr>
<td>StringIds</td>
<td>true</td>
</tr>
<tr>
<td>UuidIds</td>
<td>false</td>
</tr>
<tr>
<td>CustomIds</td>
<td>false</td>
</tr>
<tr>
<td>AnyIds</td>
<td>false</td>
</tr>
<tr>
<td>Vertex Property Feature</td>
<td>Enabled</td>
</tr>
<tr>
<td>UserSuppliedIds</td>
<td>false</td>
</tr>
<tr>
<td>AddProperty</td>
<td>true</td>
</tr>
<tr>
<td>RemoveProperty</td>
<td>true</td>
</tr>
<tr>
<td>NumericIds</td>
<td>true</td>
</tr>
<tr>
<td>StringIds</td>
<td>true</td>
</tr>
<tr>
<td>UuidIds</td>
<td>false</td>
</tr>
<tr>
<td>CustomIds</td>
<td>false</td>
</tr>
<tr>
<td>AnyIds</td>
<td>false</td>
</tr>
</tbody>
</table>
Properties

<table>
<thead>
<tr>
<th>Value</th>
<th>Neptune Implementation</th>
<th>Amazon Neptune Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>SerializableValues</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>UniformListValues</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>BooleanArrayValues</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>DoubleArrayValues</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>IntegerArrayValues</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>StringArrayValues</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>BooleanValues</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>ByteValues</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>DoubleValues</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>FloatValues</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>IntegerValues</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>LongValues</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>MapValues</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>MixedListValues</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>StringValues</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>ByteArrayValues</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>FloatArrayValues</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>LongArrayValues</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>

Edge Feature

<table>
<thead>
<tr>
<th>Feature</th>
<th>Neptune Implementation</th>
<th>Amazon Neptune Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddEdges</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>RemoveEdges</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>UserSuppliedIds</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>AddProperty</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>RemoveProperty</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>NumericIds</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>StringIds</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>UuidIds</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>CustomIds</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>AnyIds</td>
<td>false</td>
<td>false</td>
</tr>
</tbody>
</table>
Using the Gremlin Console to Connect to a Neptune DB Instance

The Gremlin Console allows you to experiment with TinkerPop graphs and queries in a REPL (read-eval-print loop) environment.

You can use the Gremlin Console to connect to a remote graph database. The following section walks you through the configuration of the Gremlin Console to connect remotely to a Neptune DB instance. You must follow these instructions from an Amazon EC2 instance in the same virtual private cloud (VPC) as your Neptune DB instance.
To connect to Neptune using the Gremlin Console

1. The Gremlin Console binaries require Java 8. Type the following to install Java 8 on your EC2 instance.
   ```
   sudo yum install java-1.8.0-devel
   ```

2. Type the following to set Java 8 as the default runtime on your EC2 instance.
   ```
   sudo /usr/sbin/alternatives --config java
   ```
   When prompted, enter the number for Java 8.

3. Download Gremlin Console (version 3.3.2+) from the Apache Tinkerpop3 website on to your EC2 instance.
   ```
   wget https://archive.apache.org/dist/tinkerpop/3.3.2/apache-tinkerpop-gremlin-console-3.3.2-bin.zip
   ```

4. Unzip the Gremlin Console zip file.
   ```
   unzip apache-tinkerpop-gremlin-console-3.3.2-bin.zip
   ```

5. Change directories into the unzipped folder.
   ```
   cd apache-tinkerpop-gremlin-console-3.3.2
   ```

6. In the `conf` subdirectory of the extracted directory, create a file named `neptune-remote.yaml` with the following text. Replace `your-neptune-endpoint` with the hostname or IP address of your Neptune DB instance. The square brackets ([ ]) are required.

   ```yaml
   hosts: [your-neptune-endpoint]
   port: 8182
   serializer: { className: org.apache.tinkerpop.gremlin.driver.ser.GryoMessageSerializerV3d0, config: { serializeResultToString: true }}
   ```

   **Note**
   For information about finding the hostname of your Neptune DB instance, see the Finding the Endpoint for a Neptune Cluster (p. 24) section.

7. In a terminal, navigate to the Gremlin Console directory (`apache-tinkerpop-gremlin-console-3.3.2`), and then type the following command to run the Gremlin Console.
   ```
   bin/gremlin.sh
   ```

   You should see the following output:
   ```
   
   \,.../
   ( o o )
   -----000o-(3)-000o-----
   plugin activated: tinkerpop.server
   plugin activated: tinkerpop.utilities
   plugin activated: tinkerpop.tinkergraph
   gremlin>
   ```

   You are now at the `gremlin>` prompt. You will type the remaining steps at this prompt.
8. At the `gremlin>` prompt, type the following to connect to the Neptune DB instance.

   `:remote connect tinkerpop.server conf/neptune-remote.yaml`

9. At the `gremlin>` prompt, type the following to switch to remote mode. This sends all Gremlin queries to the remote connection.

   `:remote console`

10. Type the following to send a query to the Gremlin Graph.

    `g.V().limit(1)`

11. When you are finished, type the following to exit the Gremlin Console.

    `:exit`

**Note**

Use a semicolon (`;`) or a newline character (`\n`) to separate each statement. Each traversal preceding the final traversal must end in `next()` to be executed. Only the data from the final traversal is returned.

For more information on the Neptune implementation of Gremlin, see the section called “Neptune Gremlin Implementation Differences” (p. 27).

For more information about Amazon Neptune, see Next Steps (p. 44).

### Using the HTTP REST Endpoint to Connect to a Neptune DB Instance

Amazon Neptune provides an HTTP endpoint for Gremlin queries. The REST interface is compatible with Gremlin version 3.3.2.

The following instructions walk you through connecting to the Gremlin endpoint using the `curl` command and HTTP. You must follow these instructions from an Amazon EC2 instance in the same virtual private cloud (VPC) as your Neptune DB instance.

The HTTP endpoint for Gremlin queries to a Neptune DB instance is `http://your-neptune-endpoint:8182/gremlin`.

**Note**

For information about finding the hostname of your Neptune DB instance, see Finding the Endpoint for a Neptune Cluster (p. 24).

**To connect to Neptune using the HTTP REST endpoint**

- The following example uses `curl` to submit a Gremlin query through HTTP POST. The query is submitted in JSON format in the body of the post as the `gremlin` property.

  ```bash
curl -X POST -d '{"gremlin":"g.V().limit(1)"}' http://your-neptune-endpoint:8182/gremlin
  ```

  **Note**
  
  Amazon Neptune does not support the `bindings` property.
You can also send queries through HTTP **GET** requests, but **HTTP POST** requests are recommended.

```bash
```

**Important**

The REST endpoint returns all results in a single JSON result set. If the result set is too large, this can cause an `OutOfMemoryError` exception on the Neptune DB instance.

For more information about the Gremlin REST interface, see Connecting via HTTP in the Apache TinkerPop3 documentation.

The preceding example returns the first vertex in the graph by using the `g.V().limit(1)` traversal. To query for something else, replace it with another Gremlin traversal.

For more information about Amazon Neptune, see Next Steps (p. 44).

### Using Java to Connect to a Neptune DB Instance

The following section walks you through the running of a complete Java sample that connects to a Neptune DB instance and performs a Gremlin traversal.

These instructions must be followed from an Amazon EC2 instance in the same virtual private cloud (VPC) as your Neptune DB instance.

**To connect to Neptune using Java**

1. Install Apache Maven on your EC2 instance. First, type the following to add a repository with a Maven package:

   ```bash
   ```

   Type the following to set the version number for the packages:

   ```bash
   sudo sed -i s/\$releasever/6/g /etc/yum.repos.d/epel.apache-maven.repo
   ```

   Then use **yum** to install Maven:

   ```bash
   sudo yum install -y apache-maven
   ```

2. The Gremlin libraries require Java 8. Type the following to install Java 8 on your EC2 instance:

   ```bash
   sudo yum install java-1.8.0-devel
   ```

3. Type the following to set Java 8 as the default runtime on your EC2 instance:

   ```bash
   sudo /usr/sbin/alternatives --config java
   ```

   When prompted, type the number for Java 8 (2).

4. Type the following to set Java 8 as the default compiler on your EC2 instance:

   ```bash
   sudo /usr/sbin/alternatives --config javac
   ```
When prompted, type the number for Java 8 (2).

5. Create a new directory named `gremlinjava`:

```
mkdir gremlinjava
cd gremlinjava
```

6. In the `gremlinjava` directory, create a `pom.xml` file, and then open it in a text editor:

```
nano pom.xml
```

7. Copy the following into the `pom.xml` file and save it:

```xml
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.amazonaws</groupId>
  <artifactId>GremlinExample</artifactId>
  <packaging>jar</packaging>
  <version>1.0-SNAPSHOT</version>
  <name>GremlinExample</name>
  <url>http://maven.apache.org</url>
  <dependencies>
    <dependency>
      <groupId>org.apache.tinkerpop</groupId>
      <artifactId>gremlin-driver</artifactId>
      <version>3.3.2</version>
    </dependency>
  </dependencies>
  <build>
    <plugins>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-compiler-plugin</artifactId>
        <version>2.0.2</version>
        <configuration>
          <source>1.8</source>
          <target>1.8</target>
        </configuration>
      </plugin>
      <plugin>
        <groupId>org.codehaus.mojo</groupId>
        <artifactId>exec-maven-plugin</artifactId>
        <version>1.3</version>
        <configuration>
          <mainClass>com.amazonaws.App</mainClass>
          <complianceLevel>1.8</complianceLevel>
        </configuration>
      </plugin>
    </plugins>
  </build>
</project>
```

**Note**

If you are modifying an existing Maven project, the required dependency is highlighted in the preceding code.

8. Create subdirectories for the example source code (`src/main/java/com/amazonaws/`) by typing the following at the command line:
9. In the `src/main/java/com/amazonaws/` directory, create a file named `App.java`, and then open it in a text editor.

```bash
cmpath src/main/java/com/amazonaws/
```

10. Copy the following into the `App.java` file. Replace `your-neptune-endpoint` with the address of your Neptune DB instance. You must NOT include the `https://` prefix in the `addContactPoint` method.

    Note
    For information about finding the hostname of your Neptune DB instance, see Finding the Endpoint for a Neptune Cluster (p. 24).

```java
package com.amazonaws;
import org.apache.tinkerpop.gremlin.driver.Cluster;
import org.apache.tinkerpop.gremlin.driver.Client;
import org.apache.tinkerpop.gremlin.driver.remote.DriverRemoteConnection;
import org.apache.tinkerpop.gremlin.process.traversal.dsl.graph.GraphTraversalSource;
import org.apache.tinkerpop.gremlin.process.traversal.dsl.graph.GraphTraversal;
import org.apache.tinkerpop.gremlin.structure.util.empty.EmptyGraph;

public class App {
    public static void main( String[] args )
    {
        Cluster.Builder builder = Cluster.build();
        builder.addContactPoint("your-neptune-endpoint");
        builder.port(8182);

        Cluster cluster = builder.create();

        GraphTraversalSource g = EmptyGraph.instance().traversal().withRemote(DriverRemoteConnection.using(cluster));

        GraphTraversal t = g.V().limit(2).valueMap();
        t.forEachRemaining(
            e -> System.out.println(e)
        );

        cluster.close();
    }
}
```

11. Compile and run the sample using the following Maven command:

```bash
mvn compile exec:java
```

The preceding example returns a map of the key and values of each property for the first two vertexes in the graph by using the `g.V().limit(2).valueMap()` traversal. To query for something else, replace it with another Gremlin traversal.

For more information about Amazon Neptune, see Next Steps (p. 44).
Using Python to Connect to a Neptune DB Instance

The following section walks you through the running of a Python sample that connects to an Amazon Neptune DB instance and performs a Gremlin traversal.

You must follow these instructions from an Amazon EC2 instance in the same virtual private cloud (VPC) as your Neptune DB instance.

Before you begin, do the following:

- Download and install Python 2.7 or later from the Python.org website.
- Verify that you have pip installed. If you don't have pip or you're not sure, see Do I need to install pip? in the pip documentation.

To connect to Neptune using Python

1. Type the following to install the gremlinpython package:
   ```bash
   pip install ##user gremlinpython
   ```

2. Create a file named gremlinexample.py, and then open it in a text editor.
3. Copy the following into the gremlinexample.py file. Replace your-neptune-endpoint with the address of your Neptune DB instance.
   ```python
   from __future__ import print_function  # Python 2/3 compatibility
   from gremlin_python import statics
   from gremlin_python.structure.graph import Graph
   from gremlin_python.process.graph_traversal import __
   from gremlin_python.process.strategies import *
   from gremlin_python.driver.driver_remote_connection import DriverRemoteConnection
   graph = Graph()
   g = graph.traversal().withRemote(DriverRemoteConnection('ws://your-neptune-endpoint:8182/gremlin','g'))
   print(g.V().limit(2).toList())
   ```

4. Type the following command to run the sample:
   ```bash
   python gremlinexample.py
   ```

   The Gremlin query at the end of this example returns the vertices (g.V().limit(2)) in a list. This list is then printed with the standard Python print function.

   **Note**
   The final part of the Gremlin query, toList(), is required to submit the traversal to the server for evaluation. If you don't include that method or another equivalent method, the query is not submitted to the Neptune DB instance.

   The following methods submit the query to the Neptune DB instance:
   - toList()
   - toSet()
• next()
• nextTraverser()
• iterate()

The preceding example returns the first two vertices in the graph by using the
`g.V().limit(2).toList()` traversal. To query for something else, replace it with another
Gremlin traversal with one of the appropriate ending methods.

For more information about Amazon Neptune, see Next Steps (p. 44).

Using .NET to Connect to a Neptune DB Instance

The following section contains a code example written in C# that connects to a Neptune DB instance and
performs a Gremlin traversal.

Connections to Amazon Neptune must be from an Amazon EC2 instance in the same virtual private cloud
(VPC) as your Neptune DB instance. This sample code was tested on an Amazon EC2 instance running
Ubuntu.

Before you begin, do the following:

• Install .NET on the Amazon EC2 instance. To get instructions for installing .NET on multiple operating
  systems, including Windows, Linux, and macOS, see Get Started with .NET.
• Install Gremlin.NET. For more information, see Gremlin.NET in the TinkerPop documentation.

To connect to Neptune using Gremlin.NET

1. Create a new .NET project.

        dotnet new console -o gremlinExample

2. Change directories into the new project directory.

        cd gremlinExample

3. Copy the following into the `Program.cs` file. Replace `your-neptune-endpoint` with the address
   of your Neptune DB instance.

   For information about finding the address of your Neptune DB instance, see the Accessing a Neptune
   Graph (p. 24) section.

```csharp
using System;
using System.Threading.Tasks;
using System.Collections.Generic;
using Gremlin.Net;
using Gremlin.Net.Driver;

namespace gremlinExample
{
    class Program
    {
        static void Main(string[] args)
        {
            try
            {
                var endpoint = "your-neptune-endpoint";
            }
        }
    }
}
```
// This uses the default Neptune and Gremlin port, 8182
var gremlinServer = new GremlinServer(endpoint);
var gremlinClient = new GremlinClient(gremlinServer);

Program program = new Program();

program.RunQueryAsync(gremlinClient).Wait();

} catch (Exception e)
{
    Console.WriteLine("0", e);
}

private async Task RunQueryAsync(GremlinClient gremlinClient)
{
    var count = await gremlinClient.SubmitWithSingleResultAsync<long>(
        "g.V().limit(1).count().next()");
    Console.WriteLine("0", count);
}

4. Type the following command to run the sample:

```
dotnet run
```

The Gremlin query at the end of this example returns the count of a single vertex for testing purposes. It is then printed to the console.

**Note**

The final part of the Gremlin query, `next()`, is required to submit the traversal to the server for evaluation. If you don't include that method or another equivalent method, the query is not submitted to the Neptune DB instance.

The following methods submit the query to the Neptune DB instance:

- `toList()`
- `toSet()`
- `next()`
- `nextTraverser()`
- `iterate()`

The preceding example returns a number by using the `g.V().limit(1).count().next()` traversal. To query for something else, replace it with another Gremlin traversal with one of the appropriate ending methods.

For more information about Amazon Neptune, see Next Steps (p. 44).

**Using Node.js to Connect to a Neptune DB Instance**

The following section walks you through the running of a Node.js sample that connects to an Amazon Neptune DB instance and performs a Gremlin traversal.

You must follow these instructions from an Amazon EC2 instance in the same virtual private cloud (VPC) as your Neptune DB instance.
Before you begin, do the following:

- Verify that Node.js is installed. If it is not, download and install Node.js from the Nodejs.org website.

## To connect to Neptune using Node.js

1. Type the following to install the `gremlin-javascript` package:

   ```bash
npm install gremlin@2.7.0
   ```

2. Create a file named `gremlinexample.js` and open it in a text editor.

3. Copy the following into the `gremlinexample.js` file. Replace `your-neptune-endpoint` with the address of your Neptune DB instance.

   For information about finding the address of your Neptune DB instance, see the Accessing a Neptune Graph (p. 24) section.

   ```javascript
gremlin = require('gremlin');
const client = gremlin.createClient(8182, "your-neptune-endpoint", { accept: "application/vnd.gremlin-v3.0+json" });
client.execute('g.V().limit(2)', (err, results) => {
  if (err) {
    console.error(err)
  } else {
    console.log(results);
    // Close the connection. Call this method only when you are done with the client.
    client.closeConnection();
  }
});
```

4. Type the following command to run the sample:

   ```bash
   node gremlinexample.js
   ```

   The preceding example returns the first two vertices in the graph by using the `g.V().limit(2)` traversal. To query for something else, replace it with another Gremlin traversal.

   **Important**

   This is a standalone Node.js example. If you are using this code in an AWS Lambda function, ensure that you indicate success or failure by using the callback parameter or the context parameter. Failure to do so can result in timeouts.

   For more information, see Using the Callback Parameter in the AWS Lambda documentation.

   The following shows the previous example with the changes that are necessary for it to run as an AWS Lambda function. This example requires that you create a deployment package that includes the `gremlin-javascript` package. For more information about creating a Lambda deployment package, see Creating a Deployment Package (Node.js).

   ```javascript
gremlin = require('gremlin');
exports.handler = (event, context, callback) => {
```
const client = gremlin.createClient(8182, "your-neptune-endpoint", { accept:
"application/vnd.gremlin-v3.0+json" });

client.execute('g.V().limit(2)', (err, results) => {
    // Close the connection
    client.closeConnection();
    if (err) {
        context.fail('error updating event: ' + err);
        return console.error(err);
    } else {
        console.log(results);
        context.succeed(results);
    }
});

For more information about Amazon Neptune, see Next Steps (p. 44).

Gremlin HTTP and WebSocket API

Gremlin HTTP requests all use a single endpoint: http://your-neptune-endpoint:8182/gremlin

Note
Amazon Neptune does not support the bindings property.

For more information about connecting to the Gremlin endpoint, see Accessing the Neptune Graph with Gremlin (p. 26).

The Amazon Neptune implementation of Gremlin has specific details and differences that you need to consider. For more information, see Neptune Gremlin Implementation Differences (p. 27).

For information about the Gremlin language and traversals, see The Traversal in the Apache TinkerPop documentation.

Next Steps

These resources provide more information about Neptune and Gremlin traversals.

- Loading Data into Neptune (p. 59)
- Accessing the Neptune Graph with SPARQL (p. 44)
- More about Gremlin queries / traversals:
  - The Graph in the Apache TinkerPop3 documentation
  - The Traversal in the Apache TinkerPop3 documentation

Accessing the Neptune Graph with SPARQL

SPARQL is a query language for the Resource Description Framework (RDF), which is a graph data format designed for the web. Amazon Neptune is compatible with SPARQL 1.1. This means that you can connect to a Neptune DB instance and query the graph using the query language described in the SPARQL 1.1 Query Language specification.

A query in SPARQL consists of a SELECT clause to specify the variables to return and a WHERE clause to specify which data to match in the graph. If you are unfamiliar with SPARQL queries, see Writing Simple Queries in the SPARQL 1.1 Query Language.
Important
Neptune does not support SPARQL UPDATE LOAD from URI. For small datasets, SPARQL UPDATE INSERT might be an option. If you need to load data from a file, see Loading Data into Neptune (p. 59).

Before you begin, you must have the following:

- A Neptune DB instance. For information about creating a Neptune DB instance, see Getting Started with Neptune (p. 17).
- An Amazon EC2 instance in the same virtual private cloud (VPC) as your Neptune DB instance.

Topics
- Using the RDF4J Console to Connect to a Neptune DB Instance (p. 45)
- Using the HTTP REST Endpoint to Connect to a Neptune DB Instance (p. 47)
- Using Java to Connect to a Neptune DB Instance (p. 47)
- Using RDF4J Workbench to Connect to a Neptune DB Instance (p. 50)
- SPARQL HTTP API (p. 51)
- SPARQL Query Status and Cancellation (p. 53)
- Next Steps (p. 55)

Using the RDF4J Console to Connect to a Neptune DB Instance

The RDF4J Console allows you to experiment with Resource Description Framework (RDF) graphs and queries in a REPL (read-eval-print loop) environment.

You can add a remote graph database as a repository and query it from the RDF4J Console. This section walks you through the configuration of the RDF4J Console to connect remotely to a Neptune DB instance.

To connect to Neptune using the RDF4J Console

1. Download the RDF4J SDK from the Download page on the RDF4J website.
2. Unzip the RDF4J SDK zip file.
3. In a terminal, navigate to the RDF4J SDK directory, and then type the following command to run the RDF4J Console:

   ```bash
   bin/console.sh
   ```

   You should see output similar to the following:

   ```
   14:11:51.126 [main] DEBUG o.e.r.c.platform.PlatformFactory - os.name = linux
   14:11:51.130 [main] DEBUG o.e.r.c.platform.PlatformFactory - Detected Posix platform
   Connected to default data directory
   RDF4J Console 2.1.5

   2.1.5
   Type 'help' for help.
   >
   ```
You are now at the > prompt. This is the general prompt for the RDF4J Console. You use this prompt for setting up repositories and other operations. A repository has its own prompt for running queries.

4. At the > prompt, type the following to create a SPARQL repository for your Neptune DB instance:

```
cREATE SPARQL
```

5. The RDF4J Console prompts you for values for the variables required to connect to the SPARQL endpoint.

```
Please specify values for the following variables:
```

Specify the following values:

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPARQL query endpoint</td>
<td><code>http://your-neptune-endpoint:8182/sparql</code></td>
</tr>
<tr>
<td>SPARQL update endpoint</td>
<td><code>http://your-neptune-endpoint:8182/sparql</code></td>
</tr>
<tr>
<td>Local repository ID</td>
<td><code>neptune</code></td>
</tr>
<tr>
<td>Repository title</td>
<td><code>Neptune DB instance</code></td>
</tr>
</tbody>
</table>

For information about finding the address of your Neptune DB instance, see the Accessing a Neptune Graph (p. 24) section.

If the operation is successful, you see the following message:

```
Repository created
```

6. At the > prompt, type the following to connect to the Neptune DB instance:

```
OPEN neptune
```

If the operation is successful, you see the following message:

```
Opened repository 'neptune'
```

You are now at the `neptune>` prompt. At this prompt, you can run queries against the Neptune graph.

**Note**

Now that you have added the repository, the next time you run `bin/console.sh`, you can immediately run the `OPEN neptune` command to connect to the Neptune DB instance.

7. At the `neptune>` prompt, type the following to run a SPARQL query that returns up to 10 of the triples (subject-predicate-object) in the graph by using the `?s ?p ?o` query with a limit of 10. To query for something else, replace the text after the `sparql` command with another SPARQL query.

```
```
For more information about Amazon Neptune, see Next Steps (p. 55).

**Using the HTTP REST Endpoint to Connect to a Neptune DB Instance**

Amazon Neptune provides an HTTP endpoint for SPARQL queries. The REST interface is compatible with SPARQL version 1.1.

The following instructions walk you through connecting to the SPARQL endpoint using the `curl` command and HTTP. You must follow these instructions from an Amazon EC2 instance in the same virtual private cloud (VPC) as your Neptune DB instance.

The HTTP endpoint for SPARQL queries to a Neptune DB instance is `http://your-neptune-endpoint:8182/sparql`.

**Note**
For information about finding the hostname of your Neptune DB instance, see the Finding the Endpoint for a Neptune Cluster (p. 24) section.

**QUERY using HTTP POST**

The following example uses `curl` to submit a SPARQL **QUERY** through HTTP **POST**.

```bash
```

The preceding example returns up to 10 of the triples (subject-predicate-object) in the graph by using the `?s ?p ?o` query with a limit of 10. To query for something else, replace it with another SPARQL query.

**Note**
The default MIME type of a response is `application/sparql-results+json` for **SELECT** and **ASK** queries.

The default MIME type of a response is `application/n-quads` for **CONSTRUCT** and **DESCRIBE** queries.

For a list of all available MIME types, see **SPARQL HTTP API (p. 51)**.

**UPDATE using HTTP POST**

The following example uses `curl` to submit a SPARQL **UPDATE** through HTTP **POST**.

```bash
```

The preceding example inserts the following triple into the SPARQL default graph: `<http://test.com/s> <http://test.com/p> <http://test.com/o>`

For more information about the Neptune SPARQL REST interface, see **SPARQL HTTP API (p. 51)**. For more information about Amazon Neptune, see **Next Steps (p. 55)**.

**Using Java to Connect to a Neptune DB Instance**

This section walks you through the running of a complete Java sample that connects to an Amazon Neptune DB instance and performs a SPARQL query.
You must follow these instructions from an Amazon EC2 instance in the same virtual private cloud (VPC) as your Neptune DB instance.

**To connect to Neptune using Java**

1. Install Apache Maven on your EC2 instance. First, type the following to add a repository with a Maven package:

   ```
sudo wget http://repos.fedorapeople.org/repos/dchen/apache-maven/epel-apache-maven.repo
   -O /etc/yum.repos.d/epel-apache-maven.repo
   ```

   Type the following to set the version number for the packages:

   ```
sudo sed -i s/\$releasever/6/g /etc/yum.repos.d/epel-apache-maven.repo
   ```

   Then you can use **yum** to install Maven:

   ```
sudo yum install -y apache-maven
   ```

2. This example was tested with Java 8 only. Type the following to install Java 8 on your EC2 instance:

   ```
sudo yum install java-1.8.0-devel
   ```

3. Type the following to set Java 8 as the default runtime on your EC2 instance:

   ```
sudo /usr/sbin/alternatives --config java
   ```

   When prompted, type the number for Java 8.

4. Type the following to set Java 8 as the default compiler on your EC2 instance:

   ```
sudo /usr/sbin/alternatives --config javac
   ```

   When prompted, type the number for Java 8.

5. In a new directory, create a **pom.xml** file, and then open it in a text editor.

6. Copy the following into the **pom.xml** file and save it:

   ```xml
   <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
     <modelVersion>4.0.0</modelVersion>
     <groupId>com.amazonaws</groupId>
     <artifactId>RDFExample</artifactId>
     <packaging>jar</packaging>
     <version>1.0-SNAPSHOT</version>
     <name>RDFExample</name>
     <url>http://maven.apache.org</url>
     <dependencies>
       <dependency>
         <groupId>org.eclipse.rdf4j</groupId>
         <artifactId>rdf4j-runtime</artifactId>
         <version>2.2</version>
       </dependency>
     </dependencies>
   </project>
   ```
7. To create subdirectories for the example source code (`src/main/java/com/amazonaws/`), type the following at the command line:

```bash
mkdir -p src/main/java/com/amazonaws/
```

8. In the `src/main/java/com/amazonaws/` directory, create a file named `App.java`, and then open it in a text editor.

9. Copy the following into the `App.java` file. Replace `your-neptune-endpoint` with the address of your Neptune DB instance.

```java
package com.amazonaws;

import org.eclipse.rdf4j.repository.Repository;
import org.eclipse.rdf4j.repository.http.HTTPRepository;
import org.eclipse.rdf4j.repository.sparql.SPARQLRepository;
import java.util.List;
import org.eclipse.rdf4j.RDF4JException;
import org.eclipse.rdf4j.repository.RepositoryConnection;
import org.eclipse.rdf4j.query.TupleQuery;
import org.eclipse.rdf4j.query.TupleQueryResult;
import org.eclipse.rdf4j.query.BindingSet;
import org.eclipse.rdf4j.query.QueryLanguage;
import org.eclipse.rdf4j.model.Value;

public class App {
    public static void main( String[] args ) {
        String sparqlEndpoint = "http://your-neptune-endpoint:8182/sparql";
        Repository repo = new SPARQLRepository(sparqlEndpoint);
        repo.initialize();

        try (RepositoryConnection conn = repo.getConnection()) {
```
TupleQuery tupleQuery = conn.prepareTupleQuery(QueryLanguage.SPARQL, queryString);

try (TupleQueryResult result = tupleQuery.evaluate()) {
  while (result.hasNext()) { // iterate over the result
    BindingSet bindingSet = result.next();

    Value s = bindingSet.getValue("s");
    Value p = bindingSet.getValue("p");
    Value o = bindingSet.getValue("o");

    System.out.print(s);
    System.out.print("\t");
    System.out.print(p);
    System.out.print("\t");
    System.out.println(o);
  }
}

10. Use the following Maven command to compile and run the sample:

mvn compile exec:java

The preceding example returns up to 10 of the triples (subject-predicate-object) in the graph by using the ?s ?p ?o query with a limit of 10. To query for something else, replace the query with another SPARQL query.

The iteration of the results in the example prints the value of each variable returned. The Value object is converted to a String and then printed. If you change the SELECT part of the query, you must modify the code.

For more information about Amazon Neptune, see Next Steps (p. 55).

Using RDF4J Workbench to Connect to a Neptune DB Instance

This section walks you through connecting to an Amazon Neptune DB instance using RDF4J Workbench and RDF4J Server. RDF4J Server is required because it acts as a proxy between the Neptune SPARQL HTTP REST endpoint and RDF4J Workbench.

RDF4J Workbench provides an easy interface for experimenting with a graph, including loading local files. For information, see the Add section in the RDF4J documentation.

Before you begin, do the following:

- Install Java 1.8 or later.
- Install RDF4J Server and RDF4J Workbench. For information, see Installing RDF4J Server and RDF4J Workbench.

To use RDF4J Workbench to connect to Neptune

1. In a web browser, navigate to the URL where the RDF4J Workbench web app is deployed. For example, if you are using Apache Tomcat, the URL is: http://ec2_hostname:8080/rdf4j-workbench/.
2. In the left pane, choose **New repository**.

   In **New repository**:
   - In the **Type** drop-down list, choose **SPARQL endpoint proxy**.
   - For **ID**, type **neptune**.
   - For **Title**, type **Neptune DB instance**.

   Choose **Next**.

3. In **New repository**:

   For information about finding the address of your Neptune DB instance, see the Accessing a Neptune Graph (p. 24) section.

   Choose **Create**.

4. The **neptune** repository now appears in the list of repositories. It might take a few minutes before you can use the new repository.

5. In the **Id** column of the table, choose the **neptune** link.

6. In the left pane, choose **Query**.

   **Note**
   - If the menu items under **Explore** are disabled, you might need to reconnect to the RDF4J Server and choose the **neptune** repository again.
   - You can do this by using the [change] links in the upper-right corner.

7. In the query field, type the following SPARQL query, and then choose **Execute**.

   ```sparql
   ```

   The preceding example returns up to 10 of the triples (subject-predicate-object) in the graph by using the ?s ?p ?o query with a limit of 10.

   For more information about Amazon Neptune, see Next Steps (p. 44).

**SPARQL HTTP API**

SPARQL HTTP requests are accepted at the following endpoint: `http://your-neptune-endpoint:8182/sparql`

For more information about connecting to Amazon Neptune with SPARQL, see Accessing the Neptune Graph with SPARQL (p. 44).

For more information about the SPARQL protocol and query language, see the SPARQL 1.1 Protocol and the SPARQL 1.1 Query Language specification.

**SPARQL UPDATE**

- **SPARQL UPDATE LOAD** from URI works only with resources that are within the same virtual private cloud (VPC).

  This includes Amazon S3 URLs in the **us-east-1** Region with an Amazon S3 VPC endpoint created. For information about creating a VPC endpoint, see Amazon S3 VPC Endpoint (p. 68).
• \texttt{SPARQL \ UPDATE \ LOAD} URI must start with \texttt{http://} or \texttt{https://}. This includes Amazon S3 URIs.

Any Amazon S3 authentication must be included in the URL. For more information, see \textit{Authenticating Requests: Using Query Parameters}.

If you need to load data from a file, we recommend that you use the Amazon Neptune loader API. For more information, see \textit{Loading Data into Neptune (p. 59)}.

\textbf{Note}

The Amazon Neptune loader API is non-ACID.

\textbf{SPARQL HTTP Content Types}

You can choose the MIME type of a SPARQL response by sending an "\texttt{Accept: type}" header with the request. For example, \texttt{curl -H "Accept: application/nquads ..."}.

The available content types depend on the SPARQL query type.

\textbf{SELECT}

• \texttt{application/sparql-results+json} \textbf{Default}
• \texttt{application/sparql-results+xml}
• \texttt{application/x-binary-rdf-results-table}
• \texttt{text/tab-separated-values}
• \texttt{text/csv}

\textbf{ASK}

• \texttt{application/sparql-results+json} \textbf{Default}
• \texttt{application/sparql-results+xml}
• \texttt{text/boolean}

\textbf{CONSTRUCT}

• \texttt{application/n-quads} \textbf{Default}
• \texttt{application/rdf+xml}
• \texttt{application/ld+json}
• \texttt{application/n-triples}
• \texttt{text/turtle}
• \texttt{text/n3}
• \texttt{application/trix}
• \texttt{application/trig}
• \texttt{application/sparql-results+json}

\textbf{DESCRIBE}

• \texttt{application/n-quads} \textbf{Default}
• \texttt{application/rdf+xml}
• \texttt{application/ld+json}
• \texttt{application/n-triples}
• \texttt{text/turtle}
SPARQL Query Status and Cancellation

Amazon Neptune provides additional APIs to check the status of running SPARQL queries and the cancellation of running SPARQL queries.

Status requests are accepted at the following endpoint: http://your-neptune-endpoint:8182/sparql/status

Topics
- SPARQL Query Status (p. 53)
- SPARQL Query Cancellation (p. 54)

SPARQL Query Status

To get the status of SPARQL queries, use HTTP GET or POST to request to the http://your-neptune-endpoint:8182/sparql/status endpoint.

Request Parameters

queryId (optional)

The ID of a running SPARQL query. Only displays the status of the specified query.

Response Syntax

```json
{
   "acceptedQueryCount": integer,
   "runningQueryCount": integer,
   "queries": [
      {
         "queryId": "guid",
         "queryEvalStats": {
            "subqueries": integer,
            "elapsed": integer,
            "cancelled": boolean
         },
         "queryString": "string"
      }
   ]
}
```

Response Values

acceptedQueryCount

The number of queries successfully submitted.

runningQueryCount

The number of currently running SPARQL queries.
queries
A list of the current SPARQL queries.

queryId
A GUID id for the query.

queryEvalStats
Statistics for this query.

subqueries
Number of subqueries in this query.

elapsed
The number of milliseconds the query has been running so far.

cancelled
True indicates the query has been cancelled.

queryString
The submitted query.

Example
The following is an example of the status command using curl and HTTP GET.

curl http://your-neptune-endpoint:8182/sparql/status

This output shows a single running query.

{
    "acceptedQueryCount":9,
    "runningQueryCount":1,
    "queries": [
        {
            "queryId":"fb34cd3e-f37c-4d12-9cf2-03bb741bf54f",
            "queryEvalStats":
            {
                "subqueries": 0,
                "elapsed": 29256,
                "cancelled": false
            },
        }
    ]
}

SPARQL Query Cancellation

To get the status of SPARQL queries, use HTTP GET or POST to request to the http://your-neptune-endpoint:8182/sparql/status endpoint.

Request Parameters

cancelQuery
Required for cancellation. Parameter has no corresponding value.

**queryId**

The ID of the running SPARQL query to cancel.

**Example**

The following is an example of the status command using `curl` to cancel a query.

```
curl http://your-neptune-endpoint:8182/sparql/status \
  --data-urlencode "cancelQuery" \
  --data-urlencode "queryId=fb34cd3e-f37c-4d12-9cf2-03bb741bf54f"
```

Successful cancellation returns HTTP 200 OK.

**Next Steps**

These resources provide more information about Neptune and SPARQL queries.

- Loading Data into Neptune (p. 59).
- More about SPARQL queries and the Resource Description Framework (RDF):
  - SPARQL HTTP API (p. 51)
  - SPARQL 1.1 Query Language

**Check the Health Status of a Neptune Cluster**

Amazon Neptune provides a mechanism to check the status of the graph database on the host. It's also a good way to confirm that you are able to connect to a cluster or instance.

To check the health of a cluster by using the `curl` command:

```
curl -G https://your-neptune-endpoint:8182/status
```

The status command returns the following if the cluster or instance is healthy:

```
{"status":"healthy"}
```

If there is a problem with the Neptune DB instance, the status command returns the HTTP 500 error code. If the host is unreachable, the request times out. Ensure that you are accessing the Neptune DB instance from within the virtual private cloud (VPC) and that your security groups allow access to the Neptune DB instance.

**Neptune Service Errors**

Amazon Neptune endpoints return the standard errors for Gremlin and SPARQL when encountered.

Neptune specific errors can also be returned from the same endpoints. This section documents Neptune error messages, codes, and recommended actions.

**Topics**

- Error Format (p. 56)
- Error Messages and Codes (p. 56)
## Error Format

Neptune error messages return a relevant HTTP error code and a JSON formatted response.

```
HTTP/1.1 400 Bad Request
x-amzn-RequestId: LDM6CJP8RMQ1FHKSC1RBVJFPPNV4KQNS05AEMF66Q9ASUAAJG
Content-Type: application/x-amz-json-1.0
Content-Length: 465
Date: Thu, 15 Mar 2017 23:56:23 GMT

{
    "requestId": "0dbcded3-a9a1-4a25-b419-828c46342e47",
    "code": "ReadOnlyViolationException",
    "detailedMessage": "The request is rejected because it violates some read-only restriction, such as a designation of a replica as read-only."
    "query": "g.addV().property('name', 'Justin')",
    "exceptionStackTrace": "...
}
```

## Error Messages and Codes

The following table contains the error code, message, and HTTP status.

It also indicates whether it is ok to retry the request. Generally, it is ok to retry the request if it may succeed on a new attempt.

<table>
<thead>
<tr>
<th>Neptune Service Error Code</th>
<th>HTTP status</th>
<th>Ok to Retry?</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>BadRequestException</td>
<td>400</td>
<td>No</td>
<td>The request could not be completed.</td>
</tr>
<tr>
<td>InternalFailureException</td>
<td>500</td>
<td>Yes</td>
<td>The request processing has failed.</td>
</tr>
<tr>
<td>CancelledByUserException500</td>
<td>500</td>
<td>Yes</td>
<td>The request processing was cancelled by an authorized client.</td>
</tr>
<tr>
<td>InvalidParameterException400</td>
<td>400</td>
<td>No</td>
<td>An invalid or out-of-range value was supplied for some input parameter.</td>
</tr>
<tr>
<td>MissingParameterException400</td>
<td>400</td>
<td>No</td>
<td>A required parameter for the specified action is not supplied.</td>
</tr>
<tr>
<td>ReadOnlyViolationException400</td>
<td>400</td>
<td>No</td>
<td>The request is rejected because it violates some read-only restriction, such as a designation of a replica as read-only.</td>
</tr>
<tr>
<td>MalformedQueryException400</td>
<td>400</td>
<td>No</td>
<td>The request is rejected because it contains a query that is syntactically incorrect.</td>
</tr>
<tr>
<td>Neptune Service Error Code</td>
<td>HTTP status</td>
<td>Ok to Retry?</td>
<td>Message</td>
</tr>
<tr>
<td>----------------------------------------------------</td>
<td>-------------</td>
<td>--------------</td>
<td>----------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>MemoryLimitExceededException</td>
<td>500</td>
<td>Yes</td>
<td>The request processing did not succeed due to lack of memory, but can be retried when the server is less busy.</td>
</tr>
<tr>
<td>TimeLimitExceededException</td>
<td>500</td>
<td>Yes</td>
<td>The request processing timed out.</td>
</tr>
<tr>
<td>UnsupportedOperationException</td>
<td>400</td>
<td>No</td>
<td>The request uses a currently unsupported feature or construct.</td>
</tr>
<tr>
<td>ConstraintViolationException</td>
<td>400</td>
<td>Yes</td>
<td>The query engine discovered, during the execution of the request, that the completion of some operation is impossible without violating some data integrity constraints, such as persistence of in- and out-vertices while adding an edge. Such conditions are typically observed if there are concurrent modifications to the graph, and are transient. The client is recommended to retry the request.</td>
</tr>
<tr>
<td>QueryLimitExceededException</td>
<td>500</td>
<td>Yes</td>
<td>The request processing did not succeed due to lack of lack of some limited resource, but can be retried when the server is less busy.</td>
</tr>
<tr>
<td>ConcurrentModificationException</td>
<td>500</td>
<td>Yes</td>
<td>The request processing did not succeed due to lack of lack of some limited resource, but can be retried when the server is less busy.</td>
</tr>
<tr>
<td>QueryTooLargeException</td>
<td>400</td>
<td>No</td>
<td>The request was rejected because its body is too large.</td>
</tr>
<tr>
<td>Neptune Service Error Code</td>
<td>HTTP status</td>
<td>Ok to Retry?</td>
<td>Message</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-------------</td>
<td>--------------</td>
<td>---------</td>
</tr>
<tr>
<td>MethodNotAllowedException</td>
<td>405</td>
<td>No</td>
<td>The request is rejected because the chosen HTTP method is not supported by the used endpoint.</td>
</tr>
<tr>
<td>AccessDeniedException</td>
<td>403</td>
<td>No</td>
<td>Authentication or authorization failure.</td>
</tr>
<tr>
<td>TooManyRequestsException</td>
<td>429</td>
<td>No</td>
<td>Rate of requests exceeds the allowed throughput.</td>
</tr>
<tr>
<td>ThrottlingException</td>
<td>500</td>
<td>Yes</td>
<td>The request was throttled because the server is overloaded with work, but can be retried when the server is less busy.</td>
</tr>
</tbody>
</table>
Loading Data into Neptune

Amazon Neptune provides a process for loading data from external files directly into a Neptune DB instance. You can use this process instead of executing a large number of INSERT statements, addVertex and addEdge steps, or other API calls.

The Neptune **Loader** command is faster, has less overhead, is optimized for large datasets, and supports both RDF (Resource Description Framework) and Gremlin data.

The following diagram shows an overview of the load process:

Here are the steps of the loading process:

1. Copy the data files to an Amazon Simple Storage Service (Amazon S3) bucket.
2. Create an IAM role with Read and List access to the bucket.
3. Create an Amazon S3 VPC endpoint.
4. Start the Neptune loader by sending a request via HTTP to the Neptune DB instance.
5. The Neptune DB instance assumes the IAM role to load the data from the bucket.

The following sections provide instructions for preparing and loading data into Neptune.

**Topics**

- Prerequisites: IAM Role and Amazon S3 Access (p. 60)
- Load Data Formats (p. 62)
- Example: Loading Data into a Neptune DB Instance (p. 67)
Loading data from an Amazon S3 bucket requires an AWS Identity and Access Management (IAM) role that has access to the bucket. Amazon Neptune assumes this role in order to load the data.

The following sections show how to create an IAM policy and an IAM role, associate the two, and then attach the role to your Neptune cluster.

Topics
- Creating an IAM Role to Allow Amazon Neptune to Access Amazon S3 resources (p. 60)
- Adding the IAM Role to an Amazon Neptune Cluster (p. 61)
- Creating the Amazon S3 VPC Endpoint (p. 61)

Note
These instructions require you to have access to the IAM console and permissions to manage IAM roles and policies. For more information, see Permissions for Working in the AWS Management Console in the IAM User Guide.

The Amazon Neptune console requires the user to have the following IAM permissions to attach the role to the Neptune cluster:

iam:GetAccountSummary on resource: *
iam:ListAccountAliases on resource: *
iam:PassRole on resource: *

Creating an IAM Role to Allow Amazon Neptune to Access Amazon S3 resources

After creating an IAM policy to allow Neptune to access Amazon S3 resources, create an IAM role and attach the IAM policy to the new IAM role. Start with an Amazon S3 role and modify it to allow Amazon Neptune.

To create an IAM role to allow Amazon Neptune to access AWS services

1. Open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane, choose Roles.
3. Choose Create role.
4. Under AWS service, choose S3.
5. Choose Next: Permissions.
6. Use the filter box to filter by the term S3 and check the box next to AmazonS3ReadOnlyAccess.

   Note
   This policy grants s3:Get* and s3:List* permissions to all buckets. Later steps restrict access to the role using the Trust policy.
   The Loader only requires s3:Get* and s3:List* permissions to the bucket you are loading from, so you can restrict these permissions by the S3 resource, as well.

7. Choose Next: Review.
8. Set Role Name to a name for your IAM role, for example: NeptuneLoadFromS3. You can also add an optional Role Description value, such as: "Allows Neptune to access S3 resources on your behalf."
9. Choose **Create Role**.
10. In the navigation pane, choose **Roles**.
11. In the **Search** field, type the name of the role you created, and choose the role when it appears in the list.
12. On the **Trust Relationships** tab, choose **Edit trust relationship**.
13. In the text field, paste the following trust policy.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "rds.amazonaws.com"
        ]
      },
      "Action": "sts: AssumeRole"
    }
  ]
}
```
14. Choose **Update trust policy**.
15. Complete the steps in **Adding the IAM Role to an Amazon Neptune Cluster** (p. 61).

### Adding the IAM Role to an Amazon Neptune Cluster

Use the console to add the IAM role to an Amazon Neptune cluster. This allows any Neptune DB instance in the cluster to assume the role and load from Amazon S3.

**Note**
The Amazon Neptune console requires the user to have the following IAM permissions to attach the role to the Neptune cluster:

- `iam:GetAccountSummary` on resource: *
- `iam:ListAccountAliases` on resource: *
- `iam:PassRole` on resource: *

**To add an IAM role to an Amazon Neptune cluster**

2. In the navigation pane, choose **Clusters**.
3. Choose the radio button next to the cluster you want to modify.
4. Under **Actions**, choose **Manage IAM roles**.
5. Choose the IAM role you created in the previous section.
6. Choose **Done**.

### Creating the Amazon S3 VPC Endpoint

The Neptune loader requires a VPC endpoint for Amazon S3.
To set up access for Amazon S3

1. Sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the left navigation pane, choose Endpoints.
3. Choose Create Endpoint.
4. Choose the Service Name com.amazonaws.region.s3.
   
   **Note**
   
   If the region here is incorrect, make sure the console region is correct.
5. Choose the VPC that contains your Neptune DB instance.
6. Select the check box next to the route tables that are associated with the subnets related to your cluster. If you only have one route table, you must select that box.
7. Choose Create Endpoint.

For information about creating the endpoint, see VPC Endpoints in the Amazon VPC User Guide. For information about the limitations of VPC endpoints, VPC Endpoints for Amazon S3.

Next Steps

Now that you have granted access to the Amazon S3 bucket, you can prepare to load data. For information about supported formats, see Load Data Formats (p. 62).

Load Data Formats

The Amazon Neptune Load API currently requires specific formats for incoming data. The following formats are available, and are listed with their identifiers for the Neptune loader API in parentheses.

- CSV format (csv) for property graph / Gremlin
- N-Triples (ntriples) format for RDF / SPARQL
- N-Quads (nquads) format for RDF / SPARQL
- RDF/XML (rdfxml) format for RDF / SPARQL
- Turtle (turtle) format for RDF / SPARQL

**Important**

All files must be encoded in UTF-8 format. If a file is not in UTF format, Neptune tries to load it anyway as UTF-8 data.

If your data is not in a supported format, you must convert it before you load it into a Neptune DB instance.

Compression Support

Neptune supports compression of single files in gzip format. The file name must end in the .gz extension and must contain a single text file encoded in UTF-8 format. Multiple files can be loaded, but each one must be contained in a separate .gz file (or uncompressed text file). Archive files (for example, .tar, .tar.gz, and .tgz) are not supported.

The following sections describe the formats in more detail.

**Topics**

- Gremlin Load Data Format (p. 63)
- RDF Load Data Formats (p. 67)
Gremlin Load Data Format

To load Apache TinkerPop Gremlin data using the CSV format, you must specify the vertices and the edges in separate files. For each load command, the set of files to be loaded must be in the same folder in the Amazon S3 bucket, and you specify the folder name for the source parameter. The file names and file name extensions are not important.

The Amazon Neptune CSV format follows the RFC 4180 CSV specification. For more information, see Common Format and MIME Type for CSV Files on the Internet Engineering Task Force (IETF) website.

**Note**

All files must be encoded in UTF-8 format.

Each file has a comma-separated header row. The header row consists of both system column headers and property column headers.

### System Column Headers

The required and allowed system column headers are different for vertex files and edge files.

Each system column can appear only once in a header.

All labels are case sensitive.

#### Vertex headers

- **~id** - *Required*
  
  An ID for the vertex.

- **~label**
  
  A label for the vertex. Multiple label values are allowed. Separate values with a semicolon (;) character.

#### Edge headers

- **~id** - *Required*
  
  An ID for the edge.

- **~from** - *Required*
  
  The vertex ID of the from vertex.

- **~to** - *Required*
  
  The vertex ID of the to vertex.

- **~label**
  
  A label for the edge. Edges can only have a single label.

### Property Column Headers

You can specify a column for a property by using the following syntax. The type names are not case sensitive.

```plaintext
propertyname:type
```
You can specify a column for an array type by adding `[]` to the type.

```
propertyname:type
```

**Note**
Spaces are not allowed in the column headers, so property names cannot include spaces.

The following example shows the column header for a property named `age` of type `Int`.

```
age:Int
```

Every row in the file would be required to have an integer in that position or be left empty.

Arrays of strings are allowed, but strings in an array must not include the semicolon (`;`) character.

The following section lists all the available data types.

## Data Types

This is a list of the allowed property types, with a description of each type.

**Bool (or Boolean)**
Indicates a Boolean field. Allowed values: 0, 1, `false`, `true`

### Whole Number Types

Values outside of the defined ranges result in an error.

<table>
<thead>
<tr>
<th>Type</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Byte</td>
<td>-127 to 126</td>
</tr>
<tr>
<td>Short</td>
<td>-32768 to 32767</td>
</tr>
<tr>
<td>Int</td>
<td>-2^31 to 2^31-1</td>
</tr>
<tr>
<td>Long</td>
<td>-2^63 to 2^63-1</td>
</tr>
</tbody>
</table>

### Decimal Number Types

Supports both decimal notation or scientific notation. Also allows symbols such as `(+-)` INFINITY or NaN. INF is not supported.

<table>
<thead>
<tr>
<th>Type</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Float</td>
<td>32-bit IEEE 754 floating point</td>
</tr>
<tr>
<td>Double</td>
<td>64-bit IEEE 754 floating point</td>
</tr>
</tbody>
</table>

Float and double values that are too long are loaded and rounded to the nearest value for 24-bit (float) and 53-bit (double) precision. A midway value is rounded to 0 for the last remaining digit at the bit level.

**String**
Quotation marks are optional. Commas, newline, and carriage return characters are automatically escaped if they are included in a string surrounded by double quotation marks ("), *Example: "Hello, World"

To include quotation marks in a quoted string, you can escape the quotation mark by using two in a row: *Example: "Hello ""World""

Arrays of strings are allowed, but strings in an array must not include the semicolon (;) character.

If you want to surround strings in an array with quotation marks, you must surround the whole array with one set of quotation marks. *Example: "String one; String 2; String 3"

**Date**

Java date in ISO-8601 format. Supports the following formats: YYYY-MM-DD, YYYY-MM-DDTHH:mm, YYYY-MM-DDTHH:mm:ss, YYYY-MM-DDTHH:mm:ssZ

**Row format**

**Delimiters**

Fields in a row are separated by a comma. Records are separated by a newline or a newline followed by a carriage return.

**Blank Fields**

Blank fields are allowed for non-required columns (such as user-defined properties). A blank field still requires a comma separator. The example in the next section has a blank field in each example vertex.

**Vertex IDs**

~id values must be unique for all vertices in every vertex file. Multiple vertex rows with identical ~id values are applied to a single vertex in the graph.

**Edge IDs**

Additionally, ~id values must be unique for all edges in every edge file. Multiple edge rows with identical ~id values are applied to the single edge in the graph.

**Labels**

Labels are case sensitive.

**String Values**

Quotation marks are optional. Commas, newline, and carriage return characters are automatically escaped if they are included in a string surrounded by double quotation marks (").

**CSV Specification**

The Neptune CSV format follows the RFC 4180 CSV specification, including the following requirements.

- Both Unix and Windows style line endings are supported (\n or \r\n).
- Any field can be quoted (using double quotation marks).
- Fields containing a line-break, double-quote, or commas must be quoted. (If they are not, load aborts immediately.)
- A double quotation mark character (") in a field must be represented by two (double) quotation mark characters. For example, a string Hello "World" must be present as "Hello ""World"" in the data.
- Surrounding spaces between delimiters are ignored. If a row is present as value1, value2, they are stored as "value1" and "value2".

API Version 2017-11-29
Gremlin Load Data Format

- Any other escape characters are stored verbatim. For example, "data1\tdata2" is stored as "data1\tdata2". No further escaping is needed as long as these characters are enclosed within quotation marks.
- Blank fields are allowed. A blank field is considered an empty value.
- Multiple values for a field are specified with a semicolon (;) between values.

For more information, see Common Format and MIME Type for CSV Files on the Internet Engineering Task Force (IETF) website.

Example

The following diagram shows an example of two vertices and an edge taken from the TinkerPop Modern Graph.

The following is the graph in Neptune CSV load format.

Vertex file:

```plaintext
~id, name:String, age:Int, lang:String, ~label
v1, "marko", 29, , person
v2, "lop", , "java", software
```

Tabular view of the vertex file:

<table>
<thead>
<tr>
<th>~id</th>
<th>name:String</th>
<th>age:Int</th>
<th>lang:String</th>
<th>~label</th>
</tr>
</thead>
<tbody>
<tr>
<td>v1</td>
<td>&quot;marko&quot;</td>
<td>29</td>
<td></td>
<td>person</td>
</tr>
<tr>
<td>v2</td>
<td>&quot;lop&quot;</td>
<td></td>
<td>&quot;java&quot;</td>
<td>software</td>
</tr>
</tbody>
</table>

Edge file:

```plaintext
~id, ~from, ~to, ~label, weight:Double
e1, v1, v2, created, 0.4
```

Tabular view of the edge file:

<table>
<thead>
<tr>
<th>~id</th>
<th>~from</th>
<th>~to</th>
<th>~label</th>
<th>weight:Double</th>
</tr>
</thead>
<tbody>
<tr>
<td>e1</td>
<td>v1</td>
<td>v2</td>
<td>created</td>
<td>0.4</td>
</tr>
</tbody>
</table>
RDF Load Data Formats

To load Resource Description Framework (RDF) data, you can use one of the following standard formats as specified by the World Wide Web Consortium (W3C):

- N-Triples (ntriples) from the specification at https://www.w3.org/TR/n-triples/
- N-Quads (nquads) from the specification at https://www.w3.org/TR/n-quads/
- RDF/XML (rdfxml) from the specification at https://www.w3.org/TR/rdf-syntax-grammar/
- Turtle (turtle) from the specification at https://www.w3.org/TR/turtle/

Important
All files must be encoded in UTF-8 format. For N-Quads and N-triples data that includes Unicode characters, \uxxxxx escape sequences are supported. However, Neptune does not support normalization. If a value is present that requires normalization, it will not match byte-to-byte during querying. For more information about normalization, see the Normalization page on Unicode.org.

Example: Loading Data into a Neptune DB Instance

This example shows how to load data into Amazon Neptune. Unless stated otherwise, you must follow these steps from an Amazon Elastic Compute Cloud (Amazon EC2) instance in the same Amazon Virtual Private Cloud (VPC) as your Neptune DB instance.

Prerequisites

Before you begin, you must have the following:

- A Neptune DB instance.

  For information about launching a Neptune DB instance, see Getting Started with Neptune (p. 17).

- An Amazon Simple Storage Service (Amazon S3) bucket to put the data files in.

  You can use an existing bucket. If you don’t have an S3 bucket, see Create a Bucket in the Amazon S3 Getting Started Guide.

- An IAM role for the Neptune DB instance to assume that has an IAM policy that allows access to the data files in the S3 bucket. The policy must grant Read and List permissions.

  For information about creating a role that has access to Amazon S3 and then associating it with a Neptune cluster, see Prerequisites: IAM Role and Amazon S3 Access (p. 60).

  Note
  The Neptune Load API needs read access to the data files only. The IAM policy doesn’t need to allow write access or access to the entire bucket.
• An Amazon S3 VPC endpoint. For more information, see the Amazon S3 VPC Endpoint (p. 68) section.

Amazon S3 VPC Endpoint

The Neptune loader requires a VPC endpoint for Amazon S3.

**To set up access for Amazon S3**

1. Sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the left navigation pane, choose **Endpoints**.
3. Choose **Create Endpoint**.
4. Choose the **Service Name** `com.amazonaws.region.s3`.
   
   **Note**
   If the region here is incorrect, make sure the console region is correct.
5. Choose the VPC that contains your Neptune DB instance.
6. Select the check box next to the route tables that are associated with the subnets related to your cluster. If you only have one route table, you must select that box.
7. Choose **Create Endpoint**.

For information about creating the endpoint, see VPC Endpoints in the Amazon VPC User Guide. For information about the limitations of VPC endpoints, VPC Endpoints for Amazon S3.

**To load data into a Neptune DB instance**

1. Copy the data files to an Amazon S3 bucket. The S3 bucket must be in the same AWS Region as the cluster that loads the data.

   You can use the following AWS CLI command to copy the files to the bucket.
   
   **Note**
   This command does not need to be run from the Amazon EC2 instance.

   ```bash
   aws s3 cp data-file-name s3://bucket-name/object-key-name
   ```
   
   **Note**
   In Amazon S3, an **object key name** is the entire path of a file, including the file name. 
   **Example**: In the command `aws s3 cp datafile.txt s3://examplebucket/mydirectory/datafile.txt`, the object key name is `mydirectory/datafile.txt`.

   Alternatively, you can use the AWS Management Console to upload files to the S3 bucket. Open the Amazon S3 console at https://console.aws.amazon.com/s3/, and choose a bucket. In the upper-left corner, choose **Upload** to upload files.
2. From a command line window, type the following to run the Neptune loader, replacing the values for the endpoint, Amazon S3 path, format, and access keys.

   The **format** parameter can be any of the following values: `csv` (Gremlin), `ntriples`, `nquads`, `turtle`, and `rdfxml` (RDF). For information about the other parameters, see Loader Command (p. 70).

   **Note**
   For information about finding the hostname of your Neptune DB instance, see the Finding the Endpoint for a Neptune Cluster (p. 24) section.
The region parameter must match the region of the cluster and the S3 bucket.

Amazon Neptune is available in the following regions:

- us-east-1 - US East (N. Virginia)
- us-east-2 - US East (Ohio)
- us-west-2 - US West (Oregon)
- eu-west-1 - EU (Ireland)

```bash
curl -X POST \
-H 'Content-Type: application/json' \nhttp://your-neptune-endpoint:8182/loader -d '{
   "source" : "s3://bucket-name/object-key-name",
   "format" : "format",
   "iamRoleArn" : "arn:aws:iam::account-id:role/role-name",
   "region" : "region",
   "failOnError" : "FALSE"
}'
```

For information about creating and associating an IAM role with a Neptune cluster, see Prerequisites: IAM Role and Amazon S3 Access (p. 60).

**Note**
The `SOURCE` parameter accepts an Amazon S3 URI that points to either a single file or a folder. If you specify a folder, Neptune loads every data file in the folder. The URI can be in any of the following formats.

- `s3://bucket_name/object-key-name`
- `https://s3.amazonaws.com/bucket_name/object-key-name`
- `https://s3-us-east-1.amazonaws.com/bucket_name/object-key-name`

3. The Neptune loader returns a job id that allows you to check the status or cancel the loading process; for example:

```json
{
   "status" : "200 OK",
   "payload" : {
      "loadId" : "ef478d76-d9da-4d94-8ff1-08d9d4863aa5"
   }
}
```

4. Type the following to get the status of the load with the `loadId` from **Step 3**:

```bash
curl -G 'http://your-neptune-endpoint:8182/loader/ef478d76-d9da-4d94-8ff1-08d9d4863aa5'
```

If the status of the load lists an error, you can request more detailed status and a list of the errors. For more information and examples, see Loader Get Status (p. 74).

5. (Optional) Cancel the Load job.

   Type the following to **Delete** the loader job with the `job id` from **Step 3**:

```bash
curl -X DELETE 'http://your-neptune-endpoint:8182/loader/ef478d76-d9da-4d94-8ff1-08d9d4863aa5'
```

The `DELETE` command returns the HTTP code 200 OK upon successful cancellation.
The data from files from the load job that has finished loading is not rolled back. The data remains in the Neptune DB instance.

Neptune Loader API Reference

This section describes the Loader APIs for Amazon Neptune that are available from the HTTP endpoint of a Neptune DB instance.

Topics
- Loader Command (p. 70)
- Loader Get Status (p. 74)
- Loader Cancel Job (p. 81)

Loader Command

Loads data from an Amazon S3 bucket into a Neptune DB instance.

To load data, you must send an HTTP POST request to the http://your-neptune-endpoint:8182/loader endpoint. The parameters for the loader request can be sent in the POST body or as URL-encoded parameters.

**Important**
The MIME type must be application/json.

The S3 bucket must be in the same AWS Region as the cluster.

**Request Syntax**

```json
{
    "source": "string",
    "format": "string",
    "iamRoleArn": "string",
    "mode": "NEW|RESUME|AUTO",
    "region": "us-east-1",
    "failOnError": "string",
    "parserConfiguration": {
        "baseUri": "http://base-uri-string",
        "namedGraphUri": "http://named-graph-string"
    }
}
```

**Request Parameters**

**source**

An Amazon S3 URI.

The source parameter accepts an Amazon S3 URI that points to either a single file or a folder. If you specify a folder, Neptune loads every data file in the folder.

The URI can be in any of the following formats.

- `s3://bucket_name/object-key-name`
- `https://s3.amazonaws.com/bucket_name/object-key-name`
• https://s3-us-east-1.amazonaws.com/bucket_name/object-key-name

**format**

The format of the data. For more information about data formats for the Neptune **Loader** command, see **Loading Data into Neptune** (p. 59).

**Allowed values**: csv (Gremlin), ntriples, nquads, rdfxml, turtle (RDF)

**iamRoleArn**

The Amazon Resource Name (ARN) for an IAM role to be assumed by the Neptune DB instance for access to the S3 bucket. For information about creating a role that has access to Amazon S3 and then associating it with a Neptune cluster, see **Prerequisites: IAM Role and Amazon S3 Access** (p. 60).

**region**

The region parameter must match the region of the cluster and the S3 bucket.

Amazon Neptune is available in the following regions:

• us-east-1 - US East (N. Virginia)
• us-east-2 - US East (Ohio)
• us-west-2 - US West (Oregon)
• eu-west-1 - EU (Ireland)

**mode**

Load job mode.

**RESUME** mode determines whether there is a previous load for the source and resumes the load if one is found. If a previous load is not found, the load is aborted. The loader avoids reloading data from a file that successfully completed in previous loads. In the special case where the previous loads from the same source completed successfully, the new load is completed with nothing reloaded.

**NEW** mode creates a new load request regardless of any previous loads. This mode may be used to reload data from a source after dropping the previously loaded data or to load new data available at the same source.

**AUTO** mode determines whether there is a previous load with the same source, and resumes the load if one is found just like **RESUME** mode. If a previous load is not found, then loads data from the source just like **NEW** mode.

**Default**: **AUTO**

**Allowed values**: **NEW**, **RESUME**, **AUTO**.

**failOnError**

Flag to toggle a complete stop on an error.

**Default**: **TRUE**

**Allowed values**: **TRUE**, **FALSE**

**parserConfiguration**

An optional object with additional parser configuration values. Each child parameter is also optional.
### Name

<table>
<thead>
<tr>
<th>Name</th>
<th>Example Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>namedGraphUri</td>
<td><a href="http://aws.amazon.com/neptune/vocab/v01/DefaultNamedGraph">http://aws.amazon.com/neptune/vocab/v01/DefaultNamedGraph</a></td>
<td>The default graph for all RDF formats when no graph is specified (for non-quads formats and NQUAD entries with no graph). Default is <a href="http://aws.amazon.com/neptune/vocab/v01/DefaultNamedGraph">http://aws.amazon.com/neptune/vocab/v01/DefaultNamedGraph</a></td>
</tr>
</tbody>
</table>

### [deprecated] accessKey

The `iamRoleArn` parameter is recommended instead. For information about creating a role that has access to Amazon S3 and then associating it with a Neptune cluster, see Prerequisites: IAM Role and Amazon S3 Access (p. 60).

An access key ID of an IAM role with access to the S3 bucket and data files.

For more information, see Access keys (access key ID and secret access key).

### [deprecated] secretKey

The `iamRoleArn` parameter is recommended instead. For information about creating a role that has access to Amazon S3 and then associating it with a Neptune cluster, see Prerequisites: IAM Role and Amazon S3 Access (p. 60).

For more information, see Access keys (access key ID and secret access key).

### Response Syntax

```json
{
    "status" : "200 OK",
    "payload" : {
        "loadId" : "guid_as_string"
    }
}
```

**200 OK**

Successfully started load job returns a 200 code.

### Errors

When an error occurs, a JSON object is returned in the BODY of the response. The `message` object contains a description of the error.

**Error 400**

Syntax errors return a 400 bad request error. The message describes the error.

**Error 500**
A valid request that cannot be processed returns a 500 internal server error. The message describes the error.

**Loader Error Messages**

The following are possible error messages from the loader with a description of the error.

**Max concurrent load limit breached (HTTP 400)**

You can only have 1 load job at a time.

**Couldn't find the AWS credential for iam_role_arn (HTTP 400)**

The credentials were not found. Verify the supplied credentials against the IAM console or AWS CLI output.

**S3 bucket not found for source (HTTP 400)**

The S3 bucket does not exist. Check the name of the bucket.

**The source source-uri does not exist/not reachable (HTTP 400)**

No matching files were found in the S3 bucket.

**Unable to connect to S3 endpoint. Provided source = source-uri and region = aws-region (HTTP 400)**

Unable to connect to Amazon S3. Region must match the cluster region. Ensure that you have a VPC endpoint. For information about creating a VPC endpoint, see Amazon S3 VPC Endpoint (p. 68).

**Bucket is not in provided region (aws-region) (HTTP 400)**

The bucket must be in the same AWS Region as your Neptune DB instance.

**Unable to perform S3 list operation (HTTP 400)**

The IAM user or role provided does not have List permissions on the bucket or the folder. Check the policy and/or the access control list (ACL) on the bucket.

**Start new load operation not permitted on a read-replica instance (HTTP 405)**

Loading is a write operation. Retry load on the read/write cluster endpoint.

**Failed to start load because of unknown error from S3 (HTTP 500)**

Amazon S3 returned an unknown error. Contact AWS Support.

**Invalid S3 access key (HTTP 400)**

Access key is invalid. Check the provided credentials.

**Invalid S3 secret key (HTTP 400)**

Secret key is invalid. Check the provided credentials.

**Examples**

**Example Request**

The following is a request sent via HTTP POST using the `curl` command. It loads a file in the Neptune CSV format. For more information, see Gremlin Load Data Format (p. 63).
Loader Get Status

Gets the status of a loader job.

To get load status, you must send an HTTP GET request to the http://your-neptune-endpoint:8182/loader endpoint. To get the status for a particular load request, you must include the loadId as a URL parameter, or the loadId can be appended to the URL path.

**Request Syntax**

GET http://your-neptune-endpoint:8182/loader?loadId=loadId

GET http://your-neptune-endpoint:8182/loader/loadId

GET http://your-neptune-endpoint:8182/loader

**Request Parameters**

loadId

The ID of the load job. If you do not specify a loadId, a list of load IDs is returned.

details

Include details beyond overall status. Default: False

*Allowed values:* TRUE, FALSE

errors

Include the list of errors. The list of errors is paged. The page and errorsPerPage parameters allow you to page through all the errors. Default: False

*Allowed values:* TRUE, FALSE
page
The error page number. Only valid with the errors parameter set to TRUE. Default: 1

Allowed values: Positive integers

errorsPerPage
The number of errors per each page. Only valid with the errors parameter set to TRUE. Default: 10

Allowed values: Positive integers

limit
The number of load ids to list. Only valid when requesting a list of load IDs by sending a GET request with no loadId specified. Default: 100

Allowed values: Positive integers, 1 - 100

Response Syntax

```
{
  "status" : "200 OK",
  "payload" : {
    "feedCount" : [ 
      { 
        "LOAD_FAILED" : number
      }
    ],
    "overallStatus" : {
      "fullUri" : "s3://bucket/key",
      "runNumber" : number,
      "retryNumber" : number,
      "status" : "string",
      "totalTimeSpent" : number,
      "totalRecords" : number,
      "totalDuplicates" : number,
      "parsingErrors" : number,
      "datatypeMismatchErrors" : number,
      "insertErrors" : number,
    },
    "failedFeeds" : [ 
      { 
        "fullUri" : "s3://bucket/key",
        "runNumber" : number,
        "retryNumber" : number,
        "status" : "string",
        "totalTimeSpent" : number,
        "totalRecords" : number,
        "totalDuplicates" : number,
        "parsingErrors" : number,
        "datatypeMismatchErrors" : number,
        "insertErrors" : number,
      }
    ],
    "errors" : { 
      "startIndex" : number,
      "endIndex" : number,
      "loadId" : "string",
      "errorLogs" : [ ]
    }
  }
}
```
200 OK

Successful status check returns a 200 code.

Errors

When an error occurs, a JSON object is returned in the BODY of the response. The message object contains a description of the error.

Error 400

An invalid loadId returns a 400 bad request error. The message describes the error.

Error 500

A valid request that cannot be processed returns a 500 internal server error. The message describes the error.

Response Fields

The following are the fields returned by the status command with descriptions.

These fields appear in the overallStatus field for all loads, and the failedFeeds for each failed feed.

**fullUri**

*Type: string*

The URI of the file or files to be loaded. In the format: `s3://bucket/key`

**runNumber**

*Type: unsigned long*

The run number of this load or feed. This is incremented when the load is restarted.

**retryNumber**

*Type: unsigned long*

The retry number of this load or feed. This is incremented when the loader retries a feed or load automatically.

**status**

*Type: string*

The returned status of the load or feed. **LOAD_COMPLETED** indicates a successful load with no problems. For a list of all load status values, see Overall and Feed Status Descriptions (p. 77).

**totalTimeSpent**

*Type: unsigned long*

The time, in seconds, spent to parse and insert data for the load or feed. This does not include the time spent fetching the list of source files.

**totalRecords**

*Type: unsigned long*
Total records loaded or attempted to load.

**totalDuplicates**
*Type: unsigned long*

The number of duplicate records encountered.

**parsingErrors**
*Type: unsigned long*

The number of parsing errors encountered.

**datatypeMismatchErrors**
*Type: unsigned long*

The number of records with a data type that did not match the given data.

**insertErrors**
*Type: unsigned long*

The number of records that were unable to be inserted due to errors.

### Error Fields

The following are the fields in the error field in the status response.

**startIndex**
*Type: unsigned long*

The index of the first included error.

**endIndex**
*Type: unsigned long*

The index of the last included error.

**loadId**
*Type: string*

The ID for the load. You can use this ID to print the errors for the load by setting the errors parameter to true.

**errorLogs**
*Type: list*

A list of error logs.

### Overall and Feed Status Descriptions

The following are possible responses returned in the status field from the loader with a description of the error.

**LOAD_NOT_STARTED**
Load has been recorded but not started.

LOAD_IN_PROGRESS
Load has started and is in progress.

LOAD_COMPLETED
Load has completed without any errors or errors within an acceptable threshold.

LOAD(CancellationTokened By_USER
Load has been cancelled by user.

LOAD(CancellationTokened Due To ERRORS
Load has been cancelled by the system due to errors.

LOAD(CancellationTokened UNEXPECTED ERROR
Load failed with an unexpected error.

LOAD_FAIIE
Load was rolled back because the error threshold was breached.

LOAD_S3_READ_ERROR
Feed failed due to intermittent or transient Amazon S3 connectivity issues. If any of the feeds receive this error, overall load status is set to LOAD_FAIIE.

LOAD_S3_ACCESS_DENIED_ERROR
Access was denied to the S3 bucket. If any of the feeds receive this error, overall load status is set to LOAD_FAIIE.

LOAD_COMMITTED W WRITE CONFLICTS
Loaded data committed with unresolved write conflicts.

The loader will try to resolve the write conflicts in separate transactions and update the feed status as the load progresses. If the final feed status is LOAD_COMMITTED W WRITE CONFLICTS, then try resuming the load and it will likely succeed without write conflicts. A write conflict is not usually related to bad input data, but duplicates in data can increase the likelihood of write conflicts.

LOAD_DATA_DEADLOCK
Load was automatically rolled back due to deadlock.

Examples

Example Request for Load Status
The following is a request sent via HTTP GET using the curl command.

```
curl -X GET 'http://your-neptune-endpoint:8182/loader/0a23728-afds-4574-a0bc-c29ce5f54802'
```

Example Response

{
"status" : "200 OK",
"payload" : {
   "feedCount" : [
      { "LOAD_FAILED" : 1 }
   ],
   "overallStatus" : {
      "datatypeMismatchErrors" : 0,
      "fullUri" : "s3://bucket/key",
      "insertErrors" : 0,
      "parsingErrors" : 5,
      "retryNumber" : 0,
      "runNumber" : 1,
      "status" : "LOAD_FAILED",
      "totalDuplicates" : 0,
      "totalRecords" : 5,
      "totalTimeSpent" : 3.0
   }
}

Example Request for Load Ids

The following is a request sent via HTTP GET using the curl command.

curl -X GET 'http://your-neptune-endpoint:8182/loader?limit=3'

Example Response

{
   "status" : "200 OK",
   "payload" : {
      "loadIds" : [
         "a2c0ce44-a44b-4517-8cd4-1dc144a8e5b5",
         "09683a01-6f37-4774-bb1b-5620d87f1931",
         "58085eb8-ceb4-4029-a3dc-3840969826b9"
      ]
   }
}

Example Request for Detailed Load Status

The following is a request sent via HTTP GET using the curl command.

curl -X GET 'http://your-neptune-endpoint:8182/loader/5a237328-af0d5-4574-a0bc-c29ce5f54802?details=true'

Example Response

{
   "status" : "200 OK",
   "payload" : {
      "failedFeeds" : [
         { "datatypeMismatchErrors" : 0,
         "fullUri" : "s3://bucket/key",
         "insertErrors" : 0,
         "parsingErrors" : 5,
         "retryNumber" : 0,
         "runNumber" : 1,
         "status" : "LOAD_FAILED",
         "totalDuplicates" : 0,
         "totalRecords" : 5,
         "totalTimeSpent" : 3.0
      }
   }
}
Example Request for Detailed Status with Load Errors

The following is a request sent via HTTP GET using the curl command.

curl -X GET 'http://your-neptune-endpoint:8182/loader/0a237328-afd5-4574-a0bc-c29ce5f54802?details=true&errors=true&page=1&errorsPerPage=3'

Example Response

```json
{
  "status": "200 OK",
  "payload": {
    "failedFeeds": [
      {
        "datatypeMismatchErrors": 0,
        "fullUri": "s3://bucket/key",
        "insertErrors": 0,
        "parsingErrors": 5,
        "retryNumber": 0,
        "runNumber": 1,
        "status": "LOAD_FAILED",
        "totalDuplicates": 0,
        "totalRecords": 5,
        "totalTimeSpent": 3.0
      }
    ],
    "feedCount": [
      {
        "LOAD_FAILED": 1
      }
    ],
    "overallStatus": {
      "datatypeMismatchErrors": 0,
      "fullUri": "s3://bucket/key",
      "insertErrors": 0,
      "parsingErrors": 5,
      "retryNumber": 0,
      "runNumber": 1,
      "status": "LOAD_FAILED",
      "totalDuplicates": 0,
      "totalRecords": 5,
      "totalTimeSpent": 3.0
    }
  }
}
```
Loader Cancel Job

Cancels a load job.

To load data, you must send an HTTP DELETE request to the http://your-neptune-endpoint:8182/loader endpoint. The loadId can be appended to the /loader URL path, or included as a variable in the URL.

Request Syntax

```
DELETE http://your-neptune-endpoint:8182/loader?loadId=loadId

DELETE http://your-neptune-endpoint:8182/loader/loadId
```

Request Parameters

**loadId**

The ID of the load job.

Response Syntax

```
no response body
```
200 OK
Successfully deleted load job returns a 200 code.

Errors
When an error occurs, a JSON object is returned in the BODY of the response. The message object contains a description of the error.

Error 400
An invalid loadId returns a 400 bad request error. The message describes the error.

Error 500
A valid request that cannot be processed returns a 500 internal server error. The message describes the error.

Loader Error Messages
The following are possible error messages from the cancel API with a description of the error.

The load with id = load_id does not exist or not active (HTTP 404)
The load was not found. Check the value of id parameter.

Load cancellation is not permitted on a read-replica instance. (HTTP 405)
Loading is a write operation. Retry load on the read/write cluster endpoint.

Examples

Example Request
The following is a request sent via HTTP DELETE using the curl command.

curl -X DELETE 'http://your-neptune-endpoint:8182/loader/0a237328-afd5-4574-a0bc-c29ce5f54802'
Amazon Neptune DB Instance Lifecycle

The lifecycle of an Amazon Neptune DB instance includes creating, modifying, maintaining, performing backups, rebooting, and deleting the instance. This section provides information about these processes.

**Topics**
- Backing Up and Restoring Amazon Neptune DB Instances (p. 84)
- Amazon Neptune DB Parameter Groups (p. 94)
- Modifying a Neptune DB Instance and Using the Apply Immediately Parameter (p. 96)
- Renaming a DB Instance (p. 99)
- Rebooting a DB Instance (p. 100)
- Deleting a DB Instance (p. 101)
Back up and restore snapshots of an Amazon Neptune DB instance

**Working with Backups**

Amazon Neptune creates and saves automated backups of your DB instance. It creates a storage volume snapshot of your DB instance, backing up the entire DB instance and not just individual databases. Neptune creates automated backups during the backup window of your DB instance. It saves the backups according to the backup retention period that you specify. If necessary, you can recover your database to any point in time during the backup retention period.

Your DB instance must be in the `ACTIVE` state for automated backups to occur. If your database is in another state, for example `STORAGE_FULL`, automated backups don't occur.

You can also back up your DB instance manually by creating a DB snapshot. For more information about creating a DB snapshot, see *Creating a Snapshot* (p. 87).

You can copy both automatic and manual DB snapshots, and share manual DB snapshots.

**Backup Storage**

Your Neptune backup storage for each AWS Region is composed of the automated backups and manual DB snapshots for that Region. Your backup storage is equivalent to the sum of the database storage for all instances in that Region. Moving a DB snapshot to another Region increases the backup storage in the destination Region.

All automated backups are deleted when you delete a DB instance. After you delete a DB instance, the automated backups can't be recovered. If you choose to have Neptune create a final DB snapshot before it deletes your DB instance, you can use that to recover your DB instance.

Manual snapshots are not deleted.

**Backup Window**

Automated backups occur daily during the preferred backup window. If the backup requires more time than allotted to the backup window, the backup continues after the window ends, until it finishes. The backup window can't overlap with the weekly maintenance window for the DB instance.

During the automatic backup window, storage I/O might be suspended briefly while the backup process initializes (typically under a few seconds). You might experience elevated latencies for a few minutes during backups for Multi-AZ deployments.

If you don't specify a preferred backup window when you create the DB instance, Neptune assigns a default 30-minute backup window. This window is selected at random from an eight-hour block of time per Region. The following table lists the time blocks for each region from which the default backups windows are assigned.
Backup Retention Period

You can set the backup retention period when you create a DB instance. If you don't set the backup retention period, the default backup retention period is seven days if you create the DB instance using the AWS Management Console. For DB clusters, the default backup retention period is one day regardless of how the DB cluster is created.

After you create a DB instance, you can modify the backup retention period. You can set the backup retention period to between 1 and 35 days. You can also set the backup retention period to 0, which disables automated backups. Manual snapshot limits (100 per AWS Region) don't apply to automated backups.

Important
An outage occurs if you change the backup retention period from 0 to a non-zero value or from a non-zero value to 0.

Disabling Automated Backups

In certain situations, you might want to disable automated backups temporarily; for example, while loading large amounts of data.

Important
We highly discourage disabling automated backups because it disables point-in-time recovery. Disabling automatic backups for a DB instance deletes all existing automated backups for the instance. If you disable and then re-enable automated backups, you can only restore starting from the time you re-enabled automated backups.

In this example, you disable automated backups for a DB instance named mydbinstance by setting the backup retention parameter to 0.

Disabling Automated Backups Using the Console

Follow these steps to use the AWS Management Console to disable automated backups immediately for your DB instance.

To disable automated backups immediately

1. Sign in to the AWS Management Console, and open the Amazon Neptune console at https://console.aws.amazon.com/neptune/home.
2. In the navigation pane, choose Instances, and then choose the DB instance that you want to modify.
3. Choose Instance actions, and then choose Modify. The Modify DB Instance page appears.
4. In the Backup section, for Backup retention Period, choose 0.
5. Choose Continue.
6. Under Scheduling of Modifications, choose Apply immediately.
7. Choose Modify DB Instance to save your changes and disable automated backups.
Enabling Automated Backups

If your DB instance doesn't have automated backups enabled, you can enable them at any time. You enable automated backups by setting the backup retention period to a positive non-zero value. When automated backups are enabled, an outage occurs and a backup is immediately created.

In this example, you enable automated backups for a DB instance named `mydbinstance` by setting the backup retention period to a positive non-zero value (in this case, 3).

Enabling Automated Backups Using the Console

Use the AWS Management Console to enable automated backups immediately for your DB instance.

**To enable automated backups immediately**

1. Sign in to the AWS Management Console, and open the Amazon Neptune console at https://console.aws.amazon.com/neptune/home.
2. In the navigation pane, choose **Instances**, and then choose the DB instance that you want to modify.
3. Choose **Instance actions**, and then choose **Modify**. The **Modify DB Instance** page appears.
4. For **Backup retention period**, choose a positive non-zero value, for example, 3.
5. Choose **Apply Immediately**.
6. Choose **Continue**.
7. Choose **Apply immediately**.
8. Choose **Modify DB Instance** to save your changes and enable automated backups.
Creating a Snapshot

Amazon Neptune creates a storage snapshot of your DB instance, backing up the entire DB instance.

When you create a DB snapshot, identify which DB instance you are going to back up. Then give your DB snapshot a name so that you can restore from it later. If you have IAM database authentication enabled, this setting is inherited from the source DB instance.

Creating a DB Snapshot Using the Console

Follow these steps to create a DB snapshot of a Neptune DB instance using the AWS Management Console.

To create a DB snapshot

1. Sign in to the AWS Management Console, and open the Amazon Neptune console at https://console.aws.amazon.com/neptune/home.
2. In the navigation pane, choose Instances.
3. Choose the radio button next to the DB instance in the Instances list.
4. Choose Instance actions, and then choose Take snapshot.

   The Take DB Snapshot dialog box appears.

5. In the Snapshot name box, type the name of the snapshot.
6. Choose Take Snapshot.
Restoring from a DB Snapshot

Neptune creates a snapshot of your DB instance, backing up the entire DB instance. You can create a DB instance by restoring from this DB snapshot. When you restore the DB instance, you provide the name of the DB snapshot to restore from, and then provide a name for the new DB instance that is created from the restore. You cannot restore from a DB snapshot to an existing DB instance; a new DB instance is created when you restore.

A new DB cluster is created for the DB instance.

Parameter Group Considerations

When you restore a DB instance, the default DB parameter group is associated with the restored instance. As soon as the restore is complete and your new DB instance is available, you must associate any custom DB parameter group used by the instance you restored from. You must apply these changes by using the console's Modify command.

Important
We recommend that you retain the parameter group for any DB snapshots you create, so that you can associate your restored DB instance with the correct parameter group.

Security Group Considerations

When you restore a DB instance, the default security group is associated with the restored instance. As soon as the restore is complete and your new DB instance is available, you must associate any custom security groups used by the instance you restored from. You must apply these changes by using the console's Modify command.

Restoring from a DB Snapshot Using the Console

To restore a DB instance from a DB snapshot

1. Sign in to the AWS Management Console, and open the Amazon Neptune console at https://console.aws.amazon.com/neptune/home.
2. In the navigation pane, choose Snapshots.
3. Choose the snapshot that you want to restore from.
4. Choose Actions, and then choose Restore Snapshot.
5. On the Restore DB Instance page, in the DB Instance Identifier box, type the name for your restored DB instance.
6. Choose Restore DB Instance.
7. If you want to restore the functionality of the DB instance to that of the DB instance that the snapshot was created from, you must modify the DB instance to use the security group. The next steps assume that your DB instance is in a virtual private cloud (VPC). If your DB instance is not in a VPC, use the Amazon Elastic Compute Cloud (Amazon EC2) console to locate the security group that you need for the DB instance.
   a. Sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
   b. In the navigation pane, choose Security Groups.
   c. Select the security group that you want to use for your DB instances. If necessary, add rules to link the security group to a security group for an EC2 instance.
Sharing a DB Snapshot or DB Cluster Snapshot

Using Neptune, you can share a manual DB snapshot or DB cluster snapshot in the following ways:

- Sharing a manual DB snapshot or DB cluster snapshot, whether encrypted or unencrypted, enables authorized AWS accounts to copy the snapshot.
- Sharing an unencrypted manual DB snapshot enables authorized AWS accounts to directly restore a DB instance from the snapshot instead of taking a copy of it and restoring from that. However, you can't restore a DB instance from a DB snapshot that is both shared and encrypted. Instead, you can make a copy of the DB snapshot and restore the DB instance from the copy.
- Sharing a manual DB cluster snapshot, whether encrypted or unencrypted, enables authorized AWS accounts to directly restore a DB cluster from the snapshot instead of taking a copy of it and restoring from that.

**Note**
To share an automated DB snapshot or DB cluster snapshot, copy it to make a manual version of it, and then share that copy.

You can share a manual snapshot with up to 20 other AWS accounts. You can also share an unencrypted manual snapshot as public, which makes the snapshot available to all AWS accounts. Take care when sharing a snapshot as public so that none of your private information is included in any of your public snapshots.

**Note**
When you restore a DB instance or DB cluster from a shared snapshot using the AWS Command Line Interface (AWS CLI) or Neptune API, you must specify the Amazon Resource Name (ARN) of the shared snapshot as the snapshot identifier.

Sharing an Encrypted Snapshot

You can share DB snapshots or DB cluster snapshots that have been encrypted "at rest" using the AES-256 encryption algorithm. To do this, you must take the following steps:

1. Share the AWS Key Management Service (AWS KMS) encryption key that was used to encrypt the snapshot with any accounts that you want to be able to access the snapshot.

You can share AWS KMS encryption keys with another AWS account by adding the other account to the KMS key policy. For details on updating a key policy, see Key Policies in the AWS KMS Developer Guide. For an example of creating a key policy, see Allowing Access to an AWS KMS Encryption Key (p. 89) later in this topic.

2. Use the AWS Management Console, AWS CLI, or Neptune API to share the encrypted snapshot with the other accounts.

These restrictions apply to sharing encrypted snapshots:

- You can't share encrypted snapshots as public.
- You can't share a snapshot that has been encrypted using the default AWS KMS encryption key of the AWS account that shared the snapshot.

**Allowing Access to an AWS KMS Encryption Key**

For another AWS account to copy an encrypted DB snapshot or DB cluster snapshot shared from your account, the account that you share your snapshot with must have access to the KMS key that encrypted the snapshot. To allow another AWS account access to an AWS KMS key, update the key policy for the
KMS key with the ARN of the AWS account that you are sharing to as a `Principal` in the KMS key policy, and then allow the `kms:CreateGrant` action.

After you have given an AWS account access to your KMS encryption key, to copy your encrypted snapshot, that AWS account must create an AWS Identity and Access Management (IAM) user if it doesn’t already have one. In addition, that AWS account must also attach an IAM policy to that IAM user that allows the IAM user to copy an encrypted DB snapshot using your KMS key. The account must be an IAM user and cannot be a root AWS account identity due to KMS security restrictions.

In the following key policy example, user 111122223333 is the owner of the KMS encryption key, and user 444455556666 is the account that the key is being shared with. This updated key policy gives the AWS account access to the KMS key by including the ARN for the root AWS account identity for user 444455556666 as a `Principal` for the policy, and by allowing the `kms:CreateGrant` action.

```json
{
    "Id": "key-policy-1",
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Allow use of the key",
            "Effect": "Allow",
            "Principal": {"AWS": [
                "arn:aws:iam::111122223333:user/KeyUser",
                "arn:aws:iam::444455556666:root"
            ]},
            "Action": ["kms:CreateGrant", "kms:Encrypt", "kms:Decrypt", "kms:ReEncrypt", "kms:GenerateDataKey*", "kms:DescribeKey"],
            "Resource": "*"
        },
        {
            "Sid": "Allow attachment of persistent resources",
            "Effect": "Allow",
            "Principal": {"AWS": [
                "arn:aws:iam::111122223333:user/KeyUser",
                "arn:aws:iam::444455556666:root"
            ]},
            "Action": ["kms:CreateGrant", "kms:ListGrants", "kms:RevokeGrant"],
            "Resource": "*",
            "Condition": {"Bool": {"kms:GrantIsForAWSResource": true}}
        }
    ]
}
```

Creating an IAM Policy to Enable Copying of the Encrypted Snapshot

Once the external AWS account has access to your KMS key, the owner of that AWS account can create a policy that allows an IAM user created for that account to copy an encrypted snapshot encrypted with that KMS key.

The following example shows a policy that can be attached to an IAM user for AWS account 444455556666 that enables the IAM user to copy a shared snapshot from AWS account 111122223333...
that has been encrypted with the KMS key c989c1dd-a3f2-4a5d-8d96-e793d082ab26 in the us-west-2 region.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowUseOfTheKey",
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey*",
                "kms:DescribeKey",
                "kms:CreateGrant",
                "kms:RetireGrant"
            ],
            "Resource": ["arn:aws:kms:us-west-2:111122223333:key/c989c1dd-a3f2-4a5d-8d96-e793d082ab26"]
        },
        {
            "Sid": "AllowAttachmentOfPersistentResources",
            "Effect": "Allow",
            "Action": [
                "kms:CreateGrant",
                "kms:ListGrants",
                "kms:RevokeGrant"
            ],
            "Resource": ["arn:aws:kms:us-west-2:111122223333:key/c989c1dd-a3f2-4a5d-8d96-e793d082ab26"],
            "Condition": {
                "Bool": {
                    "kms:GrantIsForAWSResource": true
                }
            }
        }
    ]
}
```

For details on updating a key policy, see Key Policies in the AWS KMS Developer Guide.

**AWS Management Console**

Using the Neptune console, you can share a manual DB snapshot or DB cluster snapshot with up to 20 AWS accounts. You can also use the console to stop sharing a manual snapshot with one or more accounts.

**To share a manual DB snapshot or DB cluster snapshot by using the Neptune console**

1. Sign in to the AWS Management Console, and open the Amazon Neptune console at https://console.aws.amazon.com/neptune/home.
2. In the navigation pane, choose Snapshots.
3. For Filter, choose Manual Snapshots.
4. Select the check box for the manual snapshot that you want to share.
5. Choose Snapshot Actions, and then choose Share Snapshot.
6. Choose one of the following options for DB Snapshot Visibility.
Sharing a Snapshot

• If the source DB cluster is unencrypted, choose Public to permit all AWS accounts to restore a DB instance from your manual DB snapshot, or choose Private to permit only AWS accounts that you specify to restore a DB instance from your manual DB snapshot.

  **Warning**
  If you set DB Snapshot Visibility to Public, all AWS accounts can restore a DB instance from your manual DB snapshot and have access to your data. Do not share any manual DB snapshots that contain private information as Public.

• If the source DB cluster is encrypted, DB Snapshot Visibility is set as Private because encrypted snapshots can't be shared as public.

7. For AWS Account ID, type the AWS account identifier for an account that you want to permit to restore a DB instance or DB cluster from your manual snapshot, and then choose Add. Repeat to include additional AWS account identifiers, up to 20 AWS accounts.

   If you make an error when adding an AWS account identifier to the list of permitted accounts, you can delete it from the list by choosing Delete at the right of the incorrect AWS account identifier.

8. After you have added identifiers for all of the AWS accounts that you want to permit to restore the manual snapshot, choose Save to save your changes.

**To stop sharing a manual DB snapshot or DB cluster snapshot with an AWS account**

1. Sign in to the AWS Management Console, and open the Amazon Neptune console at https://console.aws.amazon.com/neptune/home.
2. In the navigation pane, choose Snapshots.
3. For Filter, choose Manual Snapshots.
4. Select the check box for the manual snapshot that you want to stop sharing.
5. Choose Snapshot Actions, and then choose Share Snapshot.
6. To remove permission for an AWS account, choose Delete for the AWS account identifier for that account from the list of authorized accounts.
7. Choose Save to save your changes.

**AWS CLI**

To share a DB snapshot, use the aws neptune modify-db-snapshot-attribute command. Use the --values-to-add parameter to add a list of the IDs for the AWS accounts that are authorized to restore the manual snapshot.

The following example permits two AWS account identifiers, 123451234512 and 123456789012, to restore the DB snapshot named manual-snapshot1, and removes the all attribute value to mark the snapshot as private.

```
aws neptune modify-db-snapshot-attribute \
  --db-snapshot-identifier manual-snapshot1 \
  --attribute-name restore \
  --values-to-add '["111122223333","444455556666"]'
```

To remove an AWS account identifier from the list, use the --values-to-remove parameter. The following example prevents AWS account ID 444455556666 from restoring the snapshot.

```
aws neptune modify-db-snapshot-attribute \
  --db-snapshot-identifier manual-snapshot1 \
  --attribute-name restore \
```
Sharing a Snapshot

--values-to-remove '"444455556666 "'

API

You can also share a manual DB snapshot or DB cluster snapshot with other AWS accounts by using the Neptune API. To do so, call the `ModifyDBSnapshotAttribute` action for DB instances, or the `ModifyDBClusterSnapshotAttribute` action for Amazon Aurora DB clusters. Specify `restore` for `AttributeName`, and use the `ValuesToAdd` parameter to add a list of the IDs for the AWS accounts that are authorized to restore the manual snapshot.

To make a manual snapshot public and restorable by all AWS accounts, use the value `all`. However, take care not to add the `all` value for any manual snapshots that contain private information that you don't want to be available to all AWS accounts. Also, don't specify `all` for encrypted snapshots, because making such snapshots public isn't supported.

To remove sharing permission for an AWS account, use the `ModifyDBSnapshotAttribute` or `ModifyDBClusterSnapshotAttribute` action with `AttributeName` set to `restore` and the `ValuesToRemove` parameter. To mark a manual snapshot as private, remove the value `all` from the values list for the `restore` attribute.

The following example permits two AWS account identifiers, 123451234512 and 123456789012, to restore the DB snapshot named `manual-snapshot1`, and removes the `all` attribute value to mark the snapshot as private.

https://neptune.us-west-2.amazonaws.com/
?Action=ModifyDBSnapshotAttribute
&AttributeName=restore
&DBSnapshotIdentifier=manual-snapshot1
&SignatureMethod=HmacSHA256&SignatureVersion=4
&ValuesToAdd.member.1=123451234512
&ValuesToAdd.member.2=123456789012
&ValuesToRemove.member.1=all
&Version=2014-10-31
&X-Amz-Algorithm=AWS4-HMAC-SHA256
&X-Amz-Credential=AKIADQKE4SARGYLE/20150922/us-west-2/neptune/aws4_request
&X-Amz-Date=20150922T220515Z
&X-Amz-SignedHeaders=content-type;host;user-agent;x-amz-content-sha256;x-amz-date
&X-Amz-Signature=ef38f1ce3dab4e1dbf113d8d2a265c67d17ece1999fffd36be85714ed36ddbb3

To list all of the AWS accounts permitted to restore a snapshot, use the `DescribeDBSnapshotAttributes` or `DescribeDBClusterSnapshotAttributes` API action.
Amazon Neptune DB Parameter Groups

You manage your database configuration in Amazon Neptune by using parameters in a DB parameter group. DB parameter groups act as a container for engine configuration values that are applied to one or more DB instances.

There are two types of DB parameter groups: DB cluster parameter groups and DB parameter groups.

- DB cluster parameter groups apply to every instance in the cluster and generally have broader settings.
- DB parameter groups apply at the instance level and generally are associated with the Neptune graph engine, such as the `neptune_query_timeout` parameter.

A default DB parameter group is used if you create a DB instance without specifying a custom DB parameter group. You can't modify the parameter settings of a default DB parameter group. You must create your own DB parameter group to change parameter settings from their default value. Not all DB engine parameters can be changed in a custom DB parameter group.

Here are some important points you should know about working with parameters in a DB parameter group:

- When you change a static parameter and save the instance DB parameter group, the parameter change takes effect after you manually reboot the DB instance.
- When you change a static parameter and save the DB cluster parameter group, the parameter change takes effect after you manually reboot every DB instance in the cluster.
- Improperly setting parameters in a DB parameter group can have unintended adverse effects, including degraded performance and system instability. Always exercise caution when modifying database parameters, and back up your data before modifying a DB parameter group. Try out your parameter group setting changes on a test DB instance before applying those changes to a production DB instance.

Editing a DB Parameter Group

1. Sign in to the AWS Management Console, and open the Amazon Neptune console at https://console.aws.amazon.com/neptune/home.
2. Choose Parameter groups in the navigation pane.
3. Choose the Name link for the DB parameter group that you want to edit.
   
   (Optional) Choose Create parameter group to create a new cluster parameter group and create the new group. Then choose the Name of the new parameter group.
   
   **Important**
   This is required if you only have the default DB cluster parameter group because the default DB cluster parameter group can't be modified.

4. Choose Edit parameters.
5. Set the value for the parameters that you want to change.
6. Choose Save changes.
7. Reboot every Neptune DB instance in the Neptune cluster.
Creating a DB Parameter Group

1. Sign in to the AWS Management Console, and open the Amazon Neptune console at https://console.aws.amazon.com/neptune/home.
2. Choose Parameter groups in the left navigation pane.
3. Choose Create DB parameter group.

   The Create DB parameter group page appears.
4. In the Type list, choose DB Parameter Group or DB Cluster Parameter Group.
5. In the Group name box, type the name of the new DB parameter group.
6. In the Description box, type a description for the new DB parameter group.
7. Choose Create.
Modifying a Neptune DB Instance and Using the Apply Immediately Parameter

Most modifications to an Amazon Neptune DB instance can be applied immediately or deferred until the next maintenance window. Some modifications, such as parameter group changes, require that you manually reboot your DB instance for the change to take effect.

**Important**
Some modifications result in an outage because Neptune must reboot your DB instance for the change to take effect. Review the impact to your database and applications before modifying your DB instance settings.

**Impact of the Apply Immediately Option**

When you modify a DB instance, you can apply the changes immediately. To apply changes immediately, you choose the **Apply immediately** option in the AWS Management Console.

If you don't choose to apply changes immediately, the changes are put into the pending modifications queue. During the next maintenance window, any pending changes in the queue are applied.

**Important**
If you choose to apply changes immediately, any changes in the pending modifications queue are also applied. If any of the pending modifications require downtime, choosing to apply changes immediately can cause unexpected downtime.

**Common Settings and Downtime Notes**

The following table contains details about which settings you can modify, when the changes can be applied, and whether the changes cause downtime for the DB instance.

**Important**
If you choose to apply changes immediately, any changes in the pending modifications queue are also applied. If any of the pending modifications require downtime, choosing to apply immediately can cause unexpected downtime.

<table>
<thead>
<tr>
<th>DB instance setting</th>
<th>When the change occurs</th>
<th>Downtime notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allocated storage</td>
<td>If <strong>Apply immediately</strong> is set to true, the change occurs immediately.</td>
<td>No downtime. Performance might be degraded during the change.</td>
</tr>
<tr>
<td></td>
<td>If <strong>Apply immediately</strong> is set to false, the change occurs during the next maintenance window.</td>
<td></td>
</tr>
<tr>
<td>Auto minor version upgrade</td>
<td>The change is applied asynchronously, as soon as possible. This setting ignores the <strong>Apply immediately</strong> setting.</td>
<td>An outage occurs if a newer minor version is available, and Neptune has enabled automatic patching for that version.</td>
</tr>
<tr>
<td>Backup retention period</td>
<td>If <strong>Apply immediately</strong> is set to true, the change occurs immediately.</td>
<td>An outage occurs if you change from 0 to a nonzero value, or from a nonzero value to 0.</td>
</tr>
<tr>
<td></td>
<td>If <strong>Apply immediately</strong> is set to false, and you change</td>
<td></td>
</tr>
<tr>
<td>DB instance setting</td>
<td>When the change occurs</td>
<td>Downtime notes</td>
</tr>
<tr>
<td>------------------------</td>
<td>----------------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>the setting from a nonzero value to another nonzero value, the change is applied</td>
<td></td>
</tr>
<tr>
<td></td>
<td>asynchronously, as soon as possible. Otherwise, the change occurs during the next</td>
<td></td>
</tr>
<tr>
<td></td>
<td>maintenance window.</td>
<td></td>
</tr>
<tr>
<td>Backup window</td>
<td>The change is applied asynchronously, as soon as possible.</td>
<td>–</td>
</tr>
<tr>
<td>DB instance class</td>
<td>If <strong>Apply immediately</strong> is set to true, the change occurs immediately.</td>
<td>An outage occurs during this change.</td>
</tr>
<tr>
<td></td>
<td>If <strong>Apply immediately</strong> is set to false, the change occurs during the next maintenance</td>
<td></td>
</tr>
<tr>
<td></td>
<td>window.</td>
<td></td>
</tr>
<tr>
<td>DB instance identifier</td>
<td>If <strong>Apply immediately</strong> is set to true, the change occurs immediately.</td>
<td>An outage occurs during this change. The DB instance is rebooted.</td>
</tr>
<tr>
<td></td>
<td>If <strong>Apply immediately</strong> is set to false, the change occurs during the next maintenance</td>
<td></td>
</tr>
<tr>
<td></td>
<td>window.</td>
<td></td>
</tr>
<tr>
<td>DB parameter group</td>
<td><strong>Note:</strong> Parameter group can only be changed for an entire cluster.</td>
<td>An outage doesn't occur during this change. However, parameter changes only</td>
</tr>
<tr>
<td></td>
<td>The parameter group change occurs immediately. However, parameter changes only occur</td>
<td>occur when you reboot the DB instance manually without failover.</td>
</tr>
<tr>
<td></td>
<td>when you reboot the DB instance manually without failover.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For more information, see [Rebooting a DB Instance](p. 100).</td>
<td></td>
</tr>
<tr>
<td>Maintenance window</td>
<td>The change occurs immediately. This setting ignores the <strong>Apply immediately</strong> setting.</td>
<td>If there are one or more pending actions that cause an outage, and the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>maintenance window is changed to include the current time, those pending</td>
</tr>
<tr>
<td></td>
<td></td>
<td>actions are applied immediately, and an outage occurs.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If you set the window to the current time, there must be at least 30 minutes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>between the current time and end of the window to ensure that any pending</td>
</tr>
<tr>
<td></td>
<td></td>
<td>changes are applied.</td>
</tr>
<tr>
<td>DB instance setting</td>
<td>When the change occurs</td>
<td>Downtime notes</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Security group</td>
<td>The change is applied asynchronously, as soon as possible. This setting ignores the Apply immediately setting.</td>
<td>–</td>
</tr>
</tbody>
</table>

API Version 2017-11-29

98
Renaming a DB Instance

You can rename an Amazon Neptune DB instance by using the AWS Management Console. Renaming a DB instance can have far-reaching effects. The following is a list of things you should know before you rename a DB instance.

- When you rename a DB instance, the endpoint for the DB instance changes because the URL includes the name you assigned to the DB instance. You should always redirect traffic from the old URL to the new one.
- When you rename a DB instance, the old DNS name that was used by the DB instance is immediately deleted, but it can remain cached for a few minutes. The new DNS name for the renamed DB instance becomes effective after about 10 minutes. The renamed DB instance is not available until the new name becomes effective.
- You can't use an existing DB instance name when you are renaming an instance.
- All Read Replicas that are associated with a DB instance remain associated with that instance after it is renamed. For example, suppose that you have a DB instance that serves your production database, and the instance has several associated Read Replicas. If you rename the DB instance and then replace it in the production environment with a DB snapshot, the DB instance that you renamed still has the Read Replicas associated with it.
- Metrics and events that are associated with the name of a DB instance are maintained if you reuse a DB instance name. For example, if you promote a Read Replica and rename it to be the name of the previous master, the events and metrics that were associated with the master are then associated with the renamed instance.
- DB instance tags remain with the DB instance, regardless of renaming.
- DB snapshots are retained for a renamed DB instance.

Renaming a DB Instance Using the Console

Follow these steps to use the AWS Management Console to rename your Neptune DB instance.

To rename a DB instance

1. Sign in to the AWS Management Console, and open the Amazon Neptune console at https://console.aws.amazon.com/neptune/home.
2. In the navigation pane, choose Instances.
3. Choose the radio button next to the DB instance that you want to rename.
4. In the Instance actions drop-down menu, choose Modify.
5. Type a new name in the DB instance identifier text box. Select Apply immediately, and then choose Continue.
6. Choose Modify DB instance to complete the change.
Rebooting a DB Instance

In some cases, if you modify an Amazon Neptune DB instance, change the DB parameter group that is associated with the instance, or change a static DB parameter in a parameter group that the instance uses, you must reboot the instance for the changes to take effect.

Rebooting a DB instance restarts the database engine service. A reboot also applies to the DB instance any changes to the associated DB parameter group that were pending. Rebooting a DB instance results in a momentary outage of the instance, during which the DB instance status is set to rebooting. If the Neptune instance is configured for Multi-AZ, the reboot might be conducted through a failover. A Neptune event is created when the reboot is completed.

If your DB instance is a Multi-AZ deployment, you can force a failover from one Availability Zone to another when you choose the Reboot option. When you force a failover of your DB instance, Neptune automatically switches to a standby replica in another Availability Zone and updates the DNS record for the DB instance to point to the standby DB instance. As a result, you must clean up and re-establish any existing connections to your DB instance.

Reboot with failover is beneficial when you want to simulate a failure of a DB instance for testing or restore operations to the original Availability Zone after a failover occurs. For more information, see High Availability (Multi-AZ). When you reboot a DB cluster, it fails over to the standby replica. Rebooting a Neptune replica does not initiate a failover.

The time required to reboot is a function of the crash recovery process. To improve the reboot time, we recommend that you reduce database activities as much as possible during the reboot process to reduce rollback activity for in-transit transactions.

In the console, the Reboot option may be disabled if the DB instance is not in the Available state. This can be due to several reasons, such as an in-progress backup, a customer-requested modification, or a maintenance window action.

Note
Rebooting the primary instance of an Amazon Neptune DB cluster also automatically reboots the Neptune replicas for that DB cluster.

Rebooting a DB Instance Using the Console

Follow these steps to reboot a DB instance in the AWS Management Console.

To reboot a DB instance

1. Sign in to the AWS Management Console, and open the Amazon Neptune console at https://console.aws.amazon.com/neptune/home.
2. In the navigation pane, choose Instances.
3. Choose the DB instance that you want to reboot.
4. Choose Instance actions, and then choose Reboot.
5. To force a failover from one AZ to another, select Reboot with failover? in the Reboot DB Instance dialog box.
6. Choose Reboot. To cancel the reboot, choose Cancel instead.
Deleting a DB Instance

You can delete an Amazon Neptune DB instance in any state and at any time. To delete a DB instance, you must specify the name of the instance and whether you want to have a final DB snapshot taken of the instance. If the DB instance that you’re deleting has a status of `Creating`, you can't have a final DB snapshot taken. If the DB instance is in a failure state with a status of `failed`, `incompatible-restore`, or `incompatible-network`, you can only delete the instance when the `SkipFinalSnapshot` parameter is set to `true`.

**Important**
If you choose not to create a final DB snapshot, you can't later restore the DB instance to its final state. When you delete a DB instance, all automated backups are deleted and cannot be recovered. Manual DB snapshots of the instance are not deleted.

When you delete all instances in a cluster, the cluster is deleted, too.

If the DB instance that you want to delete has a Read Replica, you should either promote the Read Replica or delete it.

In the following examples, you delete a DB instance both with and without a final DB snapshot.

**Deleting a DB Instance with No Final Snapshot**

If you want to quickly delete a DB instance, you can skip creating a final DB snapshot. When you delete a DB instance, all automated backups are deleted and cannot be recovered. Manual snapshots are not deleted.

**Deleting a DB Instance Using the Console**

Follow these steps to use the AWS Management Console to delete a Neptune DB instance without a final DB snapshot.

**To delete a DB instance with no final DB snapshot**

1. Sign in to the AWS Management Console, and open the Amazon Neptune console at `https://console.aws.amazon.com/neptune/home`.
2. Choose `Instances` in the left navigation pane.
3. In the `Instances` list, choose the radio button next to the DB instance that you want to delete.
4. Choose `Instance actions`, and then choose `Delete` from the menu.
5. Choose `No` in the `Create final snapshot?` box.
6. Choose `Delete`.

**Deleting a DB Instance with a Final Snapshot**

If you want to be able to restore a deleted DB instance at a later time, you can create a final DB snapshot. All automated backups are also deleted and cannot be recovered. Manual snapshots are not deleted.

**Deleting a DB Instance Using the Console**

Follow these steps to use the AWS Management Console to delete a Neptune DB instance with a final DB snapshot.
To delete a DB instance with a final DB snapshot

1. Sign in to the AWS Management Console, and open the Amazon Neptune console at https://console.aws.amazon.com/neptune/home.
2. Choose **Instances** in the left navigation pane.
3. In the **Instances** list, choose the radio button next to the DB instance that you want to delete.
4. Choose **Instance actions**, and then choose **Delete** from the menu.
5. Choose **Yes** in the **Create final snapshot?** box.
6. In the **Final snapshot name** box, type the name of your final DB snapshot.
7. Choose **Delete**.
Encrypting Neptune Resources

Amazon Neptune encrypted instances use the 256-bit Advanced Encryption Standard (AES-256) to encrypt your data on the server that hosts your Neptune instance. After your data is encrypted, Neptune handles authentication of access and decryption of your data transparently with a minimal impact on performance. You don't need to modify your database client applications to use encryption.

Neptune encrypted instances provide an additional layer of data protection by helping to secure your data from unauthorized access to the underlying storage. You can use Neptune encryption to increase data protection of your applications that are deployed in the cloud, and to fulfill compliance requirements for data-at-rest encryption.

To manage the keys used for encrypting and decrypting your Neptune resources, you use AWS Key Management Service (AWS KMS). AWS KMS combines secure, highly available hardware and software to provide a key management system scaled for the cloud. Using AWS KMS, you can create encryption keys and define the policies that control how these keys can be used. AWS KMS supports AWS CloudTrail, so you can audit key usage to verify that keys are being used appropriately. You can use your AWS KMS keys in combination with Neptune and supported AWS services such as Amazon Simple Storage Service (Amazon S3), Amazon Elastic Block Store (Amazon EBS), and Amazon Redshift. For a list of services that support AWS KMS, see How AWS Services Use AWS KMS in the AWS Key Management Service Developer Guide.

All logs, backups, and snapshots are encrypted for a Neptune encrypted instance.

Enabling Encryption for a Neptune DB Instance

To enable encryption for a new Neptune DB instance, choose Yes in the Enable encryption section on the Neptune console. For information about creating a Neptune DB instance, see Getting Started with Neptune (p. 17).

When you create an encrypted Neptune DB instance, you can also supply the AWS KMS key identifier for your encryption key. If you don't specify an AWS KMS key identifier, Neptune uses your default Amazon RDS encryption key (aws/rds) for your new Neptune DB instance. AWS KMS creates your default encryption key for Neptune for your AWS account. Your AWS account has a different default encryption key for each AWS Region.

After you create an encrypted Neptune DB instance, you can't change the encryption key for that instance. So, be sure to determine your encryption key requirements before you create your encrypted Neptune DB instance.

You can use the Amazon Resource Name (ARN) of a key from another account to encrypt a Neptune DB instance. If you create a Neptune DB instance with the same AWS account that owns the AWS KMS encryption key that's used to encrypt that new Neptune DB instance, the AWS KMS key ID that you pass can be the AWS KMS key alias instead of the key's ARN.

Important
If Neptune loses access to the encryption key for a Neptune DB instance—for example, when Neptune access to a key is revoked—the encrypted DB instance is placed into a terminal state and can only be restored from a backup. We strongly recommend that you always enable backups for encrypted Neptune DB instances to guard against the loss of encrypted data in your databases.
Monitoring Neptune with CloudWatch

Amazon Neptune and Amazon CloudWatch are integrated, so you can gather and analyze performance metrics. You can monitor these metrics using the CloudWatch console, the AWS Command Line Interface (AWS CLI), or the CloudWatch API.

CloudWatch also lets you set alarms so that you can be notified if a metric value breaches a threshold that you specify. You can even set up CloudWatch Events to take corrective action if a breach occurs. For more information about using CloudWatch and alarms, see the CloudWatch Documentation.

Topics

- Viewing CloudWatch Data (Console) (p. 105)
- Viewing CloudWatch Data (AWS CLI) (p. 105)
- Viewing CloudWatch Data (API) (p. 106)
- Neptune Metrics (p. 106)
- Neptune Dimensions (p. 108)

Viewing CloudWatch Data (Console)

To view CloudWatch data for a Neptune cluster (console)

1. Sign in to the AWS Management Console and open the CloudWatch console at https://console.aws.amazon.com/cloudwatch/.
2. In the navigation pane, click Metrics.
3. In the All Metrics pane, choose Neptune, and then choose DBClusterIdentifier. Then in the upper pane, scroll down to view the full list of metrics for your cluster.

   The available Neptune metric options appear in the Viewing list.

To select or deselect an individual metric, in the results pane, select the check box next to the resource name and metric. Graphs showing the metrics for the selected items appear at the bottom of the console. To learn more about CloudWatch graphs, see Graph Metrics in the Amazon CloudWatch User Guide.

Viewing CloudWatch Data (AWS CLI)

To view CloudWatch data for a Neptune cluster (AWS CLI)

1. Install the AWS CLI. For instructions, see the AWS Command Line Interface User Guide.
2. Use the AWS CLI to fetch information. The relevant CloudWatch parameters for Neptune are listed in Neptune Metrics (p. 106).
The following example retrieves CloudWatch metrics for the number of Gremlin requests per second for the `gremlin-cluster` cluster.

```
aws cloudwatch get-metric-statistics \
  --namespace AWS/Neptune --metric-name GremlinRequestsPerSec \
  --dimensions Name=DBClusterIdentifier,Value=gremlin-cluster \
  --start-time 2018-03-03T00:00:00Z --end-time 2018-03-04T00:00:00Z \
  --period 60 --statistics=Average
```

Viewing CloudWatch Data (API)

CloudWatch also supports a `Query` action, so you can request information programmatically. For more information, see the [CloudWatch Query API documentation](https://docs.aws.amazon.com/AmazonCloudWatch/latest/APIReference/API_Query.html) and [Amazon CloudWatch API Reference](https://docs.aws.amazon.com/CloudWatch/latest/APIReference/API_Query.html).

When a CloudWatch action requires a parameter that is specific to Neptune monitoring, such as `MetricName`, use the values listed in [Neptune Metrics](#).

The following example shows a low-level CloudWatch request, using the following parameters:

- `Statistics.member.1` = `Average`
- `Dimensions.member.1` = `DBClusterIdentifier=gremlin-cluster`
- `Namespace` = `AWS/Neptune`
- `StartTime` = `2013-11-14T00:00:00Z`
- `EndTime` = `2013-11-16T00:00:00Z`
- `Period` = `60`
- `MetricName` = `GremlinRequestsPerSec`

Here is what the CloudWatch request looks like. However, this is just to show the form of the request; you must construct your own request based on your metrics and timeframe.

```http
http://monitoring.amazonaws.com/
?SignatureVersion=2
&Action=GremlinRequestsPerSec
&Version=2010-08-01
&StartTime=2018-03-03T00:00:00
&EndTime=2018-03-04T00:00:00
&Period=60
&MetricName=GremlinRequestsPerSec
&Timestamp=2018-03-04T17%3A48%3A21.746Z
&WASAccessKeyId=<AWS Access Key ID>
&Signature=<Signature>
```

Neptune Metrics

The following metrics are available from Amazon Neptune. Neptune sends metrics to CloudWatch only when they have a non-zero value.
Note
For all Neptune metrics, the aggregation granularity is five minutes.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPUUtilization</td>
<td>The percentage of CPU utilization.</td>
</tr>
<tr>
<td>ClusterReplicaLag</td>
<td>For a read replica, the amount of lag when replicating updates from the primary instance, in milliseconds.</td>
</tr>
<tr>
<td>ClusterReplicaLagMaximum</td>
<td>The maximum amount of lag between the primary instance and each Neptune DB instance in the DB cluster, in milliseconds.</td>
</tr>
<tr>
<td>ClusterReplicaLagMinimum</td>
<td>The minimum amount of lag between the primary instance and each Neptune DB instance in the DB cluster, in milliseconds.</td>
</tr>
<tr>
<td>EngineUptime</td>
<td>The amount of time that the instance has been running, in seconds.</td>
</tr>
<tr>
<td>FreeableMemory</td>
<td>The amount of available random access memory, in bytes</td>
</tr>
<tr>
<td>GremlinErrors</td>
<td>Number of errors in Gremlin traversals.</td>
</tr>
<tr>
<td>GremlinRequests</td>
<td>Number of requests to Gremlin engine.</td>
</tr>
<tr>
<td>GremlinRequestsPerSec</td>
<td>Number of requests to Gremlin engine per second.</td>
</tr>
<tr>
<td>Http100</td>
<td>Number of HTTP 100 errors for the endpoint.</td>
</tr>
<tr>
<td>Http101</td>
<td>Number of HTTP 101 errors for the endpoint.</td>
</tr>
<tr>
<td>Http200</td>
<td>Number of HTTP 200 errors for the endpoint.</td>
</tr>
<tr>
<td>Http400</td>
<td>Number of HTTP 400 errors for the endpoint.</td>
</tr>
<tr>
<td>Http403</td>
<td>Number of HTTP 403 errors for the endpoint.</td>
</tr>
<tr>
<td>Http405</td>
<td>Number of HTTP 405 errors for the endpoint.</td>
</tr>
<tr>
<td>Http413</td>
<td>Number of HTTP 413 errors for the endpoint.</td>
</tr>
<tr>
<td>Http429</td>
<td>Number of HTTP 429 errors for the endpoint.</td>
</tr>
<tr>
<td>Http500</td>
<td>Number of HTTP 500 errors for the endpoint.</td>
</tr>
<tr>
<td>Http501</td>
<td>Number of HTTP 501 errors for the endpoint.</td>
</tr>
<tr>
<td>LoaderErrors</td>
<td>Number of errors from Loader requests.</td>
</tr>
<tr>
<td>LoaderRequests</td>
<td>Number of Loader Requests.</td>
</tr>
<tr>
<td>NetworkReceiveThroughput</td>
<td>The incoming (Receive) network traffic on the DB instance, including both customer database traffic and Neptune traffic used for monitoring and replication, in bytes/second.</td>
</tr>
<tr>
<td>NetworkThroughput</td>
<td>The amount of network throughput both received from and transmitted to clients by each instance</td>
</tr>
</tbody>
</table>
### Neptune Dimensions

The metrics for Neptune are qualified by the values for the account, graph name, or operation. You can use the CloudWatch console to retrieve Neptune data along with any of the dimensions in the following table.

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBClusterIdentifier</td>
<td>Filters the data you request for a specific database instance within a cluster.</td>
</tr>
<tr>
<td>DBClusterIdentifier,Role</td>
<td>Filters the data you request for a specific Neptune DB cluster, aggregating the metric by instance role (WRITER/READER). For example, you can aggregate metrics for all READER instances that belong to a cluster.</td>
</tr>
<tr>
<td>DatabaseClass</td>
<td>Filters the data you request for all instances in a database class. For example, you can aggregate</td>
</tr>
</tbody>
</table>
### Neptune Dimensions

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>metrics for all instances that belong to the database class db.r4.large</td>
</tr>
<tr>
<td>DBClusterIdentifier, EngineName</td>
<td>Filters the data by the cluster. The engine name for all Neptune instances is neptune.</td>
</tr>
<tr>
<td>EngineName</td>
<td>The engine name for all Neptune instances is neptune.</td>
</tr>
</tbody>
</table>
Logging Amazon Neptune API Calls Using AWS CloudTrail

AWS CloudTrail is a service that logs AWS API calls made by or on behalf of your AWS account. The logging information is stored in an Amazon Simple Storage Service (Amazon S3) bucket. You can use the information collected by CloudTrail to monitor activity for your Amazon Neptune DB instances. For example, you can determine whether a request was completed successfully and which user made the request. To learn more about CloudTrail, see the AWS CloudTrail User Guide.

Important
Amazon Neptune console, AWS CLI, and API calls are logged as calls made to the Amazon Relational Database Service (Amazon RDS) API.

If an action is taken on behalf of your AWS account using the Neptune console or the Neptune command line interface, AWS CloudTrail logs the action as calls made to the Amazon RDS API. For example, if you use the Neptune console to modify a DB instance, or call the AWS CLI modify-db-instance command, the AWS CloudTrail log shows a call to the Amazon RDS API ModifyDBInstance action. For a list of the Neptune API actions that are logged by AWS CloudTrail, see the Neptune API Reference.

Note
AWS CloudTrail only logs events for Neptune API calls. If you want to audit actions taken on your database that are not part of the Neptune API, such as when a user connects to your database or when a change is made to your database schema, you must use the monitoring capabilities provided by your DB engine.

Configuring CloudTrail Event Logging

CloudTrail creates audit trails in each AWS Region separately and stores them in an Amazon S3 bucket. You can configure CloudTrail to use Amazon Simple Notification Service (Amazon SNS) to notify you when a log file is created, but that's optional. CloudTrail notifies you frequently, so we recommend that you use Amazon SNS together with an Amazon Simple Queue Service (Amazon SQS) queue and handle notifications programmatically.

You can enable CloudTrail using the AWS Management Console, AWS CLI, or API. When you enable CloudTrail logging, you can have the CloudTrail service create an Amazon S3 bucket for you to store your log files. For details, see Creating and Updating Your Trail in the AWS CloudTrail User Guide. The AWS CloudTrail User Guide also contains information about how to aggregate CloudTrail logs from multiple Regions into a single Amazon S3 bucket.

There is no cost to use the CloudTrail service. However, standard rates for Amazon S3 usage apply in addition to rates for Amazon SNS usage should you include that option. For pricing details, see the Amazon S3 and Amazon SNS pricing pages.

Neptune Event Entries in CloudTrail Log Files

CloudTrail log files contain event information formatted using JSON. An event record represents a single AWS API call. It includes information about the requested action, the user that requested the action, the date and time of the request, and so on.

CloudTrail log files include events for all AWS API calls for your account, and not just calls to the Neptune API. However, you can read the log files and scan for calls to the Neptune API using the eventName element.
The following example shows a CloudTrail log for a user that created a snapshot of a DB instance and then deleted that instance using the Neptune console. The console is identified by the userAgent element. The requested API calls made by the console (CreateDBSnapshot and DeleteDBInstance) are found in the eventName element for each record. Information about the user (Alice) can be found in the userIdentity element.

```
{
  Records:[
    {
      "awsRegion":"us-west-2",
      "eventName":"CreateDBSnapshot",
      "eventSource":"rds.amazonaws.com",
      "eventTime":"2014-01-14T16:23:49Z",
      "eventVersion":"1.0",
      "sourceIPAddress":"192.0.2.01",
      "userAgent":"AWS Console, aws-sdk-java\unknown-version Linux\2.6.18-kaos_fleet-1108-prod.2 Java_HotSpot(TM)_64-Bit_Server_VM\24.45-b08",
      "userIdentity":{
        "accessKeyId":"AKIADQKE4SARGYLE",
        "accountId":"123456789012",
        "arn":"arn:aws:iam::123456789012:user/Alice",
        "principalId":"AIDA12JXM4FBZEXAMPLE",
        "sessionContext":{
          "attributes":{
            "creationDate":"2014-01-14T15:55:59Z",
            "mfaAuthenticated":false
          }
        },
        "type":"IAMUser",
        "userName":"Alice"
      }
    },
    {
      "awsRegion":"us-west-2",
      "eventName":"DeleteDBInstance",
      "eventSource":"rds.amazonaws.com",
      "eventVersion":"1.0",
      "sourceIPAddress":"192.0.2.01",
      "userAgent":"AWS Console, aws-sdk-java\unknown-version Linux\2.6.18-kaos_fleet-1108-prod.2 Java_HotSpot(TM)_64-Bit_Server_VM\24.45-b08",
      "userIdentity":{
        "accessKeyId":"AKIADQKE4SARGYLE",
        "accountId":"123456789012",
        "arn":"arn:aws:iam::123456789012:user/Alice",
        "principalId":"AIDA12JXM4FBZEXAMPLE",
        "sessionContext":{
          "attributes":{
            "creationDate":"2014-01-14T15:55:59Z",
            "mfaAuthenticated":false
          }
        },
        "type":"IAMUser",
        "userName":"Alice"
      }
    }
  ]
}
```
For more information about the different elements and values in CloudTrail log files, see CloudTrail Log Event Reference in the AWS CloudTrail User Guide.

You might also want to use one of the AWS Partner solutions that integrate with CloudTrail to read and analyze your CloudTrail log files. For options, see the AWS CloudTrail Partners page.
Tagging Amazon Neptune Resources

You can use Amazon Neptune tags to add metadata to your Neptune resources. In addition, you can use tags with AWS Identity and Access Management (IAM) policies to manage access to Neptune resources and control what actions can be applied to those resources. Finally, you can use tags to track costs by grouping expenses for similarly tagged resources.

All Neptune resources can be tagged, including the following:

- DB instances
- DB clusters
- Read Replicas
- DB snapshots
- DB cluster snapshots
- Event subscriptions
- DB parameter groups
- DB cluster parameter groups
- DB subnet groups

Overview of Neptune Resource Tags

An Amazon Neptune tag is a name-value pair that you define and associate with a Neptune resource. The name is referred to as the key. Supplying a value for the key is optional. You can use tags to assign arbitrary information to a Neptune resource. You can use a tag key, for example, to define a category, and the tag value might be an item in that category. For example, you might define a tag key of "project" and a tag value of "Salix," indicating that the Neptune resource is assigned to the Salix project. You can also use tags to designate Neptune resources as being used for test or production by using a key such as environment=test or environment=production. We recommend that you use a consistent set of tag keys to make it easier to track metadata that is associated with Neptune resources.

Use tags to organize your AWS bill to reflect your own cost structure. To do this, sign up to get your AWS account bill with tag key values included. Then, to see the cost of combined resources, organize your billing information according to resources with the same tag key values. For example, you can tag several resources with a specific application name, and then organize your billing information to see the total cost of that application across several services. For more information, see Cost Allocation and Tagging in About AWS Billing and Cost Management.

Each Neptune resource has a tag set, which contains all the tags that are assigned to that Neptune resource. A tag set can contain as many as 10 tags, or it can be empty. If you add a tag to a Neptune resource that has the same key as an existing tag on resource, the new value overwrites the old value.

AWS does not apply any semantic meaning to your tags; tags are interpreted strictly as character strings. Neptune can set tags on a DB instance or other Neptune resources, depending on the settings that you use when you create the resource. For example, Neptune might add a tag indicating that a DB instance is for production or for testing.

- The tag key is the required name of the tag. The string value can be from 1 to 128 Unicode characters in length and cannot be prefixed with "aws: " or "rds: ". The string can contain only the set of Unicode letters, digits, white space, "_", ",", "/", ",=", ",\"", ",'", ",-\"\" (Java regex: "^[\p{L}\p{Z}\p{N}_.:/=+\-
\-]*$\").
• The tag value is an optional string value of the tag. The string value can be from 1 to 256 Unicode characters in length and cannot be prefixed with "aws:". The string can contain only the set of Unicode letters, digits, white space, ',', ':', '/', '=' or '+' (Java regex: "^([\p{L}\p{Z}\p{N}_.:/=+\-]+)$").

Values do not have to be unique in a tag set and can be null. For example, you can have a key-value pair in a tag set of project/Trinity and cost-center/Trinity.

Note
You can add a tag to a snapshot. However, your bill won't reflect this grouping.

You can use the AWS Management Console, the AWS CLI, or the Neptune API to add, list, and delete tags on Neptune resources. When using the AWS CLI or the Neptune API, you must provide the Amazon Resource Name (ARN) for the Neptune resource that you want to work with. For more information about constructing an ARN, see Constructing an ARN for Neptune (p. 117).

Tags are cached for authorization purposes. Because of this, additions and updates to tags on Neptune resources can take several minutes before they are available.

Copying Tags

When you create or restore a DB instance, you can specify that the tags from the DB instance are copied to snapshots of the DB instance. Copying tags ensures that the metadata for the DB snapshots matches that of the source DB instance, and that any access policies for the DB snapshot also match those of the source DB instance. Tags are not copied by default.

You can specify that tags are copied to DB snapshots for the following actions:

• Creating a DB instance.
• Restoring a DB instance.
• Creating a Read Replica.
• Copying a DB snapshot.

Note
If you include a value for the --tag-key parameter of the create-db-snapshot AWS CLI command (or supply at least one tag to the CreateDBSnapshot API action), Amazon Neptune doesn't copy tags from the source DB instance to the new DB snapshot. This functionality applies even if the source DB instance has the --copy-tags-to-snapshot (CopyTagsToSnapshot) option enabled. If you take this approach, you can create a copy of a DB instance from a DB snapshot and avoid adding tags that don't apply to the new DB instance.

After you create your DB snapshot using the AWS CLI create-db-snapshot command (or the CreateDBSnapshot Neptune API action), you can then add tags as described later in this topic.

AWS Management Console

The process to tag an Amazon Neptune resource is similar for all resources. The following procedure shows how to tag a Neptune DB instance.

To add a tag to a DB instance

1. Sign in to the AWS Management Console, and open the Amazon Neptune console at https://console.aws.amazon.com/neptune/home.
2. In the navigation pane, choose Instances.
Note
To filter the list of DB instances in the Instances pane, type a text string in the Filter instances box. Only DB instances that contain the string appear.

3. Choose the DB instance that you want to tag.
4. Choose Instance actions, and then choose See details.
5. In the details section, scroll down to the Tags section.
7. Type a value for Tag key and Value.
8. To add another tag, you can choose Add another Tag and type a value for its Tag key and Value.

Repeat this step as many times as necessary.

To delete a tag from a DB instance
1. Sign in to the AWS Management Console, and open the Amazon Neptune console at https://console.aws.amazon.com/neptune/home.
2. In the navigation pane, choose Instances.

Note
To filter the list of DB instances in the Instances pane, type a text string in the Filter instances box. Only DB instances that contain the string appear.

3. Choose the DB instance that you want to tag.
4. Choose Instance actions, and then choose See details.
5. In the details section, scroll down to the Tags section.
6. Choose the tag that you want to delete.
7. Choose Remove, and then choose Remove in the Remove tags window.

AWS CLI

You can add, list, or remove tags for a DB instance using the AWS CLI.

• To add one or more tags to a Neptune resource, use the AWS CLI command add-tags-to-resource.
• To list the tags on a Neptune resource, use the AWS CLI command list-tags-for-resource.
• To remove one or more tags from a Neptune resource, use the AWS CLI command remove-tags-from-resource.

To learn more about how to construct the required Amazon Resource Name (ARN), see Constructing an ARN for Neptune (p. 117).

API

You can add, list, or remove tags for a DB instance using the Neptune API.

• To add a tag to a Neptune resource, use the AddTagsToResource operation.
• To list tags that are assigned to a Neptune resource, use the ListTagsForResource.
• To remove tags from a Neptune resource, use the RemoveTagsFromResource operation.
To learn more about how to construct the required ARN, see Constructing an ARN for Neptune (p. 117).

When working with XML using the Neptune API, tags use the following schema:

```xml
<Tagging>
  <TagSet>
    <Tag>
      <Key>Project</Key>
      <Value>Trinity</Value>
    </Tag>
    <Tag>
      <Key>User</Key>
      <Value>Jones</Value>
    </Tag>
  </TagSet>
</Tagging>
```

The following table provides a list of the allowed XML tags and their characteristics. Values for Key and Value are case-dependent. For example, project=Trinity and PROJECT=Trinity are two distinct tags.

<table>
<thead>
<tr>
<th>Tagging Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TagSet</td>
<td>A tag set is a container for all tags that are assigned to a Neptune resource. There can be only one tag set per resource. You work with a TagSet only through the Neptune API.</td>
</tr>
<tr>
<td>Tag</td>
<td>A tag is a user-defined key-value pair. There can be from 1 to 50 tags in a tag set.</td>
</tr>
<tr>
<td>Key</td>
<td>A key is the required name of the tag. The string value can be from 1 to 128 Unicode characters in length and cannot be prefixed with &quot;rds:&quot; or &quot;aws:&quot;. The string can contain only the set of Unicode letters, digits, white space, ‘<em>’, ‘,’ ‘/’, ‘=’, ‘+’, ‘-’ (Java regex: &quot;^([\p{L}\p{Z}\p{N}</em>.:/=+-]*)$&quot;). Keys must be unique to a tag set. For example, you can’t have a key-pair in a tag set with the key the same but with different values, such as project/Trinity and project/Xanadu.</td>
</tr>
<tr>
<td>Value</td>
<td>A value is the optional value of the tag. The string value can be from 1 to 256 Unicode characters in length and cannot be prefixed with &quot;rds:&quot; or &quot;aws:&quot;. The string can contain only the set of Unicode letters, digits, white space, ‘<em>’, ‘,’ ‘/’, ‘=’, ‘+’, ‘-’ (Java regex: &quot;^([\p{L}\p{Z}\p{N}]</em>[.:/=+-]*)$&quot;). Values don’t have to be unique in a tag set and can be null. For example, you can have a key-value pair in a tag set of project/Trinity and cost-center/Trinity.</td>
</tr>
</tbody>
</table>

**Working with ARNs in Neptune**

Resources that are created in Amazon Web Services are each uniquely identified with an Amazon Resource Name (ARN). For certain Amazon Neptune operations, you must uniquely identify a Neptune resource by specifying its ARN.

**Important**

Amazon Neptune shares the format of Amazon RDS ARNs. Neptune ARNs contain rds and not neptune.
Constructing an ARN for Neptune

You can construct an ARN for an Amazon Neptune resource using the following syntax. Neptune shares the format of Amazon RDS ARNs.

```
arn:aws:rds:<region>:<account number>:<resourcetype>:<name>
```

The following table shows the format that you should use when constructing an ARN for a particular Neptune resource type.

<table>
<thead>
<tr>
<th>Resource Type</th>
<th>ARN Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB instance</td>
<td><code>arn:aws:rds:&lt;region&gt;:&lt;account&gt;:db:&lt;name&gt;</code></td>
</tr>
<tr>
<td></td>
<td>For example:</td>
</tr>
<tr>
<td>DB cluster</td>
<td><code>arn:aws:rds:&lt;region&gt;:&lt;account&gt;:cluster:&lt;name&gt;</code></td>
</tr>
<tr>
<td></td>
<td>For example:</td>
</tr>
<tr>
<td></td>
<td>For example:</td>
</tr>
<tr>
<td>DB parameter group</td>
<td><code>arn:aws:rds:&lt;region&gt;:&lt;account&gt;:pg:&lt;name&gt;</code></td>
</tr>
<tr>
<td></td>
<td>For example:</td>
</tr>
<tr>
<td>DB cluster parameter group</td>
<td><code>arn:aws:rds:&lt;region&gt;:&lt;account&gt;:cluster-pg:&lt;name&gt;</code></td>
</tr>
<tr>
<td></td>
<td>For example:</td>
</tr>
<tr>
<td>DB snapshot</td>
<td><code>arn:aws:rds:&lt;region&gt;:&lt;account&gt;:snapshot:&lt;name&gt;</code></td>
</tr>
<tr>
<td></td>
<td>For example:</td>
</tr>
</tbody>
</table>
Getting an Existing ARN

You can get the ARN of a Neptune resource by using the AWS Management Console, AWS Command Line Interface (AWS CLI), or Neptune API.

AWS Management Console

To get an ARN using the console, navigate to the resource that you want an ARN for, and view the details for that resource. For example, to get the ARN for a DB instance, choose Instances in the navigation panel, and choose the instance that you want from the list. The ARN is in the Instance Details section.

AWS CLI

To use the AWS CLI to get an ARN for a particular Neptune resource, use the describe command for that resource. The following table shows each AWS CLI command and the ARN property that is used with the command to get an ARN.

<table>
<thead>
<tr>
<th>AWS CLI Command</th>
<th>ARN Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>describe-event-subscriptions</td>
<td>EventSubscriptionArn</td>
</tr>
<tr>
<td>describe-certificates</td>
<td>CertificateArn</td>
</tr>
<tr>
<td>describe-db-parameter-groups</td>
<td>DBParameterGroupArn</td>
</tr>
<tr>
<td>describe-db-cluster-parameter-groups</td>
<td>DBClusterParameterGroupArn</td>
</tr>
<tr>
<td>describe-db-instances</td>
<td>DBInstanceArn</td>
</tr>
<tr>
<td>describe-db-snapshots</td>
<td>DBSnapshotArn</td>
</tr>
<tr>
<td>describe-events</td>
<td>SourceArn</td>
</tr>
<tr>
<td>describe-db-subnet-groups</td>
<td>DBSubnetGroupArn</td>
</tr>
<tr>
<td>describe-db-clusters</td>
<td>DBClusterArn</td>
</tr>
<tr>
<td>describe-db-cluster-snapshots</td>
<td>DBClusterSnapshotArn</td>
</tr>
</tbody>
</table>
For example, the following AWS CLI command gets the ARN for a DB instance.

**Example**

For Linux, OS X, or Unix:

```bash
aws neptune describe-db-instances \
--db-instance-identifier DBInstanceIdentifier \
--region us-west-2
```

For Windows:

```bash
aws neptune describe-db-instances ^
--db-instance-identifier DBInstanceIdentifier ^
--region us-west-2
```

**API**

To get an ARN for a particular Neptune resource, call the following Amazon RDS API actions, and use the ARN properties shown.

<table>
<thead>
<tr>
<th>Neptune API Action</th>
<th>ARN Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>DescribeEventSubscriptions</td>
<td>EventSubscriptionArn</td>
</tr>
<tr>
<td>DescribeCertificates</td>
<td>CertificateArn</td>
</tr>
<tr>
<td>DescribeDBParameterGroups</td>
<td>DBParameterGroupArn</td>
</tr>
<tr>
<td>DescribeDBClusterParameterGroups</td>
<td>DBClusterParameterGroupArn</td>
</tr>
<tr>
<td>DescribeDBInstances</td>
<td>DBInstanceArn</td>
</tr>
<tr>
<td>DescribeDBSnapshots</td>
<td>DBSnapshotArn</td>
</tr>
<tr>
<td>DescribeEvents</td>
<td>SourceArn</td>
</tr>
<tr>
<td>DescribeDBSubnetGroups</td>
<td>DBSubnetGroupArn</td>
</tr>
<tr>
<td>DescribeDBClusters</td>
<td>DBClusterArn</td>
</tr>
<tr>
<td>DescribeDBClusterSnapshots</td>
<td>DBClusterSnapshotArn</td>
</tr>
</tbody>
</table>
Using Audit Logs with Amazon Neptune Cluster

To audit Amazon Neptune DB cluster activity, enable the collection of audit logs by setting a DB cluster parameter. When audit logs are enabled, you can use it to log any combination of supported events. You can view or download the audit logs to review them.

Enabling Audit Logs

Use the parameter described in this section to enable and configure audit logs for your DB cluster.

Use the `neptune_enable_audit_log` parameter to enable (1) or disable (0) audit logs.

Configure audit logs by setting these parameters in the parameter group that is used by your DB cluster. You can use the procedure shown in Editing a DB Parameter Group (p. 94) to modify DB cluster parameters using the AWS Management Console. To modify DB cluster parameters programmatically, use the `modify-db-cluster-parameter-group` AWS CLI command or the `ModifyDBClusterParameterGroup` API command.

Modifying these parameters doesn't require a DB cluster restart.

Viewing Audit Logs

You can view and download the audit logs by using the AWS Management Console. On the Instances page, choose the DB instance to show its details, and then scroll to the Logs section.

To download a log file, select that file in the Logs section, and then choose Download.

Audit Log Details

Log files are in UTF-8 format. Logs are written in multiple files, the number of which varies based on the instance size. To see the latest events, you might have to review all the audit log files.

Log entries are not in sequential order. You can use the `timestamp` value for ordering.

Log files are rotated when they reach 100 MB in aggregate. This limit is not configurable.

The audit log files include the following comma-delimited information in rows, in the specified order:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timestamp</td>
<td>The Unix time stamp for the logged event with microsecond precision.</td>
</tr>
<tr>
<td>ServerHost</td>
<td>The hostname or IP of the instance that the event is logged for.</td>
</tr>
<tr>
<td>ClientHost</td>
<td>The hostname or IP that the user connected from.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ConnectionType</td>
<td>The connection type. Can be Websocket or HTTP.</td>
</tr>
<tr>
<td>RequestMessage</td>
<td>The raw request message that was sent to the Neptune endpoint.</td>
</tr>
</tbody>
</table>

Timestamp, DB server hostname, DB client hostname, Connection Type (Websocket or HTTP), and the raw request message.
IAM Database Authentication for Neptune

You can authenticate to your Neptune DB instance or DB cluster using AWS Identity and Access Management (IAM) database authentication.

When IAM database authentication is enabled, each request must be signed using AWS Signature Version 4.

AWS Signature Version 4 is the process to add authentication information to AWS requests. For security, all requests to Neptune DB clusters with IAM authentication enabled must be signed with an access key, which consists of an access key ID and secret access key. The authentication is managed externally using IAM roles.

Topics
- Enabling IAM Authentication (p. 122)
- Creating and Using an IAM Policy for IAM Database Access (p. 123)
- Attaching an IAM Policy to an IAM User or Role (p. 125)
- IAM Policy Limitations (p. 125)
- Connecting and Signing with AWS Signature Version 4 (p. 126)
- Connect to Neptune Using the Gremlin Console with Version 4 Signing (p. 126)
- Connect to Neptune Using Java and Gremlin with Version 4 Signing (p. 128)
- Connect to Neptune Using Java and SPARQL with Version 4 Signing (RDF4J and Jena) (p. 129)
- Connect to Neptune Using Python with Version 4 Signing (Example) (p. 130)
- Appendix: Prerequisites on Amazon Linux EC2 (p. 136)

Enabling IAM Authentication

By default, IAM database authentication is disabled when you create a Neptune DB cluster. You can enable IAM database authentication (or disable it again) using the AWS Management Console.

AWS Management Console

To create a new Neptune DB cluster with IAM authentication by using the console, you must follow the instructions to create a Neptune DB cluster in the Launching a Neptune DB Cluster (p. 20) topic.

On the second page of the creation process, choose Yes for Enable IAM DB Authentication.

To enable or disable IAM authentication for an existing DB instance or cluster

1. Sign in to the AWS Management Console, and open the Amazon Neptune console at https://console.aws.amazon.com/neptune/home.
2. In the navigation pane, choose Clusters.
3. Choose the Neptune DB cluster that you want to modify, and then choose Cluster actions, and then choose Modify cluster.
4. In the **Database options** section, for **IAM DB Authentication**, choose **Enable IAM DB authorization** or **No** (to disable), and then choose **Continue**.
5. To apply the changes immediately, choose **Apply immediately**.
6. Choose **Modify cluster**.

## Creating and Using an IAM Policy for IAM Database Access

To allow an IAM user or role to connect to your Neptune DB cluster, you must create an IAM policy. After that, you attach the policy to an IAM user or role.

**Note**
The IAM policy, IAM user, and Neptune DB cluster must be in the same account. Cross-account access is not supported.

The following example policy allows an IAM user to connect to Neptune DB cluster using IAM database authentication.

```json
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "neptune-db:*"
         ],
         "Resource": [
            "arn:aws:neptune-db:us-east-1:123456789012:cluster-ABCD1234EFGH5678IJKL90MNOP/*
         ]
      }
   ]
}
```

**Important**
The `neptune-db:` prefix and the `neptune-db:*` action are only for IAM database authentication. They aren't valid in any other context.

The example policy includes a single statement with the following elements:

- **Effect**—Specify **Allow** to grant access to the DB cluster. If you don't explicitly allow access, then access is denied by default.
- **Action**—Specify `neptune-db:*` to allow connection to the DB cluster.
- **Resource**—Specify an Amazon Resource Name (ARN) that describes a specific DB cluster. The ARN format is as follows:

```
arn:aws:neptune-db:region:account-id:cluster-resource-id/*
```

In this format, the following are so:

- **region** is the AWS Region for the Amazon Neptune DB cluster. In the example policy, the AWS Region is `us-east-1`.
- **account-id** is the AWS account number for the DB cluster. In the example policy, the account number is `123456789012`.
• **cluster-resource-id** is the identifier for the DB cluster. In the example policy, the identifier is `cluster-ABCD1234EFGH5678IJKL90MNOP`.

To find a cluster resource ID in the AWS Management Console for Amazon Neptune, choose the DB cluster you want and the **Resource ID** is shown in the **Details** section.

**Important**
Changes to an IAM policy take up to 10 minutes to apply to the specified Neptune resources. IAM policies applied to a Neptune DB cluster apply to all instances in that cluster.

**Grant Access to All Clusters**
The following policy uses the "*" character to match all of the DB clusters for a particular AWS account and AWS Region.

```json
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "neptune-db:*"
         ],
         "Resource": [
            "arn:aws:neptune-db:us-east-1:123456789012:/*/*"
         ]
      }
   ]
}
```

The IAM user or role has access to everything on the Neptune DB cluster. Neptune does not support fine-grained access control.

**Deny Access to a Specific Cluster**
The following policy denies access to a DB cluster for a particular AWS account and AWS Region.

The default IAM action is to deny access to a DB cluster unless an **Allow Effect** is granted.

To ensure that access is blocked, you can use the **Deny Effect**. The explicit Deny Effect takes precedent over any Allow Effect.

```json
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Deny",
         "Action": [
            "neptune-db:*"
         ],
         "Resource": [
            "arn:aws:neptune-db:us-east-1:123456789012:cluster-ABCD1234EFGH5678IJKL90MNOP/*"
         ]
      }
   ]
}
```
The IAM user or role is denied access to everything on the Neptune DB cluster. Neptune does not support fine-grained access control.

**Deny Access to All Clusters**

The following policy denies access to all DB clusters for a particular AWS account and AWS Region.

The default IAM action is to deny access to a DB cluster unless an **Allow Effect** is granted.

To ensure that access is blocked, you can use the **Deny Effect**. The explicit **Deny Effect** takes precedent over any **Allow Effect**.

```json
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Deny",
         "Action": [
            "neptune-db:*"
         ],
         "Resource": [
            "arn:aws:neptune-db:us-east-1:123456789012:/*/*"
         ]
      }
   ]
}
```

The IAM user or role is denied access to everything on the Neptune DB cluster. Neptune does not support fine-grained access control.

Next, you need to attach the IAM policy to a user or role.

---

**Attaching an IAM Policy to an IAM User or Role**

After you create an IAM policy to allow database authentication, you need to attach the policy to an IAM user or role. For a tutorial on this topic, see [Create and Attach Your First Customer Managed Policy](#) in the [IAM User Guide](#).

As you work through the tutorial, you can use one of the policy examples shown in this section as a starting point and tailor it to your needs. At the end of the tutorial, you have an IAM user with an attached policy that can use the `neptune-db:*` action.

**IAM Policy Limitations**

Changes to an IAM policy take up to 10 minutes to apply to the specified Neptune resources.

IAM policies applied to a Neptune DB cluster apply to all instances in that cluster.

Neptune does not support the following:

- AWS Global and IAM Condition Context Keys.
- Fine-grained access control.
- Actions other than `neptune-db:*`.
- Cross-account access.
Connecting and Signing with AWS Signature Version 4

Amazon Neptune resources that have IAM DB authentication enabled require all HTTP requests to be signed using AWS Signature Version 4.

For general information about signing requests with AWS Signature Version 4, see Signature Version 4 Signing Process.

AWS Signature Version 4 is the process to add authentication information to AWS requests. For security, most requests to AWS must be signed with an access key, which consists of an access key ID and secret access key.

Important
Accessing Neptune with IAM-based authentication requires that you create HTTP requests and sign the requests yourself.

How AWS Signature Version 4 works

1. You create a canonical request.
2. You use the canonical request and some other information to create a string to sign.
3. You use your AWS secret access key to derive a signing key, and then use that signing key and the string to sign to create a signature.
4. You add the resulting signature to the HTTP request in a header or as a query string parameter.

When Neptune receives the request, it performs the same steps that you did to calculate the signature. Neptune then compares the calculated signature to the one you sent with the request. If the signatures match, the request is processed. If the signatures don't match, the request is denied.

For general information about signing requests with AWS Signature Version 4, see Signature Version 4 Signing Process.

The following sections show examples that illustrate how to send signed requests to the Gremlin and SPARQL endpoints of a Neptune DB instance with IAM authentication enabled.

Topics

- the section called “Gremlin Console” (p. 126)
- the section called “Gremlin Java” (p. 128)
- the section called “SPARQL Java (RDF4J and Jena)” (p. 129)
- the section called “Connect to Neptune Using Python with Version 4 Signing (Example)” (p. 130)
- the section called “Prerequisites on EC2” (p. 136)

Connect to Neptune Using the Gremlin Console with Version 4 Signing

This section shows how to connect to Neptune using the Gremlin Console with Signature Version 4 authentication.

Prerequisites

- Java 8 or higher.
• Apache Maven 3.3 or higher.
• For information on installing these prerequisites on an EC2 instance running Amazon Linux, see Appendix: Prerequisites on Amazon Linux EC2 (p. 136).

To connect to Neptune using the Gremlin Console with Version 4 signing

1. Download Gremlin Console (version 3.3.2+) from the Apache Tinkerpop3 website on to your EC2 instance.
2. Unzip the Gremlin Console zip file.
   
   ```
   unzip apache-tinkerpop-gremlin-console-3.3.2-bin.zip
   ```
3. Clone the sample repository from GitHub.
   
   ```
   git clone https://github.com/aws/amazon-neptune-gremlin-java-sigv4.git
   ```
4. Build, package, and get the dependencies of the project.
   
   ```
   mvn package dependency:copy-dependencies
   ```
5. Copy the package jar and all dependency jars to the Gremlin Console library directory.
   
   ```
   cp target/{*.jar,dependency/*.jar} ../apache-tinkerpop-gremlin-console-3.3.2/lib
   ```
6. Change directories into the unzipped Gremlin Console directory.
   
   ```
   cd ../apache-tinkerpop-gremlin-console-3.3.2-bin
   ```
7. In the `conf` subdirectory of the extracted directory, create a file named `neptune-remote.yaml` with the following text. Replace `your-neptune-endpoint` with the hostname or IP address of your Neptune DB instance. The square brackets ([ ]) are required.
   
   **Note**
   For information about finding the hostname of your Neptune DB instance, see the Finding the Endpoint for a Neptune Cluster (p. 24) section.

   ```
   hosts: [your-neptune-endpoint]
   port: 8182
   connectionPool: {
     channelizer: org.apache.tinkerpop.gremlin.driver.SigV4WebSocketChannelizer
   }
   serializer: {
     className: org.apache.tinkerpop.gremlin.driver.ser.GryoMessageSerializerV3d0,
     config: {
       serializeResultToString: true
     }
   }
   ```
8. In a terminal, navigate to the Gremlin Console directory (apache-tinkerpop-gremlin-console-3.3.1-bin), and then type the following command to run the Gremlin Console.
   
   ```
   bin/gremlin.sh
   ```

   You should see the following output:

   ```
   \,.../ 
   (... o o) 
   ----o000o-(3)-o000---- 
   plugin activated: tinkerpop.server 
   plugin activated: tinkerpop.utilities 
   plugin activated: tinkerpop.tinkergraph 
   ```
You are now at the `gremlin>` prompt. You will type the remaining steps at this prompt.

9. At the `gremlin>` prompt, type the following to connect to the Neptune DB instance.

```bash
:remote connect tinkerpop.server conf/neptune-remote.yaml
```

10. At the `gremlin>` prompt, type the following to switch to remote mode. This sends all Gremlin queries to the remote connection.

```bash
:remote console
```

## Connect to Neptune Using Java and Gremlin with Version 4 Signing

This section shows how to connect to Neptune using the Gremlin Java API with Signature Version 4 authentication.

### Prerequisites

- Java 8 or higher.
- Apache Maven 3.3 or higher.
- For information about installing these prerequisites on an Amazon EC2 instance running Amazon Linux, see Appendix: Prerequisites on Amazon Linux EC2 (p. 24).

### To connect to Neptune using the Gremlin Java API with Version 4 signing

1. Clone the sample repository from GitHub.

   ```bash
git clone https://github.com/aws/amazon-neptune-gremlin-java-sigv4.git
```

2. Change into the cloned directory.

   ```bash
cd amazon-neptune-gremlin-java-sigv4
```

3. Type the following command to compile and run the example code.

   ```bash
mvn compile exec:java \
-Dexec.mainClass=com.amazon.neptune.gremlin.driver.example.NeptuneGremlinSigV4Example \
-Dexec.args="-e your-neptune-endpoint -p 8182"
```

   Replace `your-neptune-endpoint` with the hostname or IP address of your Neptune DB instance. The default port is 8182.

   **Note**
   For information about finding the hostname of your Neptune DB instance, see the Finding the Endpoint for a Neptune Cluster (p. 24) section.

4. To view the source code for the example, see the `NeptuneGremlinSigV4Example.java` file in the `src/main/java/com/amazon/neptune/gremlin/driver/example` directory.
To use the SigV4 signing driver in your own Java application, add the following to the `<dependencies>` section of your `pom.xml`.

```xml
<dependency>
  <groupId>org.apache.tinkerpop</groupId>
  <artifactId>gremlin-driver</artifactId>
  <version>3.3.1</version>
</dependency>
<dependency>
  <groupId>com.amazonaws</groupId>
  <artifactId>aws-java-sdk-core</artifactId>
  <version>1.11.307</version>
</dependency>
<dependency>
  <groupId>com.amazonaws</groupId>
  <artifactId>amazon-neptune-sigv4-signer</artifactId>
  <version>1.0</version>
</dependency>
<dependency>
  <groupId>com.amazonaws</groupId>
  <artifactId>amazon-neptune-gremlin-java-sigv4</artifactId>
  <version>1.0</version>
</dependency>
```

## Connect to Neptune Using Java and SPARQL with Version 4 Signing (RDF4J and Jena)

This section shows how to connect to Neptune using either RDF4J or Apache Jena with Signature Version 4 authentication.

### Prerequisites

- Java 8 or higher.
- Apache Maven 3.3 or higher.
- For information about installing these prerequisites on an EC2 instance running Amazon Linux, see [Appendix: Prerequisites on Amazon Linux EC2](#).

### To connect to Neptune using either RDF4J or Apache Jena with Version 4 signing

1. Clone the sample repository from GitHub.

   ```bash
git clone https://github.com/aws/amazon-neptune-sparql-java-sigv4.git
   ```

2. Change into the cloned directory.

   ```bash
cd amazon-neptune-sparql-java-sigv4
   ```

3. Type one of the following commands to compile and run the example code.

   Replace `your-neptune-endpoint` with the hostname or IP address of your Neptune DB instance. The default port is 8182.

   ```bash
   Note
   For information about finding the hostname of your Neptune DB instance, see the Finding the Endpoint for a Neptune Cluster (p. 24) section.
   ```
Eclipse RDF4J

Type the following to run the RDF4J example.

```
mvn compile exec:java \
    -Dexec.mainClass="com.amazonaws.neptune.client.rdf4j.NeptuneRdf4JSigV4Example" \
    -Dexec.args="http://your-neptune-endpoint:8182"
```

Apache Jena

Type the following to run the RDF4J example.

```
mvn compile exec:java \
    -Dexec.mainClass="com.amazonaws.neptune.client.jena.NeptuneJenaSigV4Example" \
    -Dexec.args="http://your-neptune-endpoint:8182"
```

4. To view the source code for the example, see the examples in the `src/main/java/com/amazonaws/neptune/client` directory.

To use the SigV4 signing driver in your own Java application, add the `amazon-neptune-sigv4-signer` Maven package to the `<dependencies>` section of your `pom.xml`. We recommend you use the examples as a starting point.

## Connect to Neptune Using Python with Version 4 Signing (Example)

This section shows an example program written in Python that illustrates how to work with Signature Version 4 for Amazon Neptune. This example is based on the examples in the Signature Version 4 Signing Process section in the Amazon Web Services General Reference.

In order to work with this example program, you need the following:

- Python 3.x installed on your computer, which you can get from the Python site. These programs were tested using Python 3.6.
- The Python `requests` library, which is used in the example script to make web requests. A convenient way to install Python packages is to use `pip`, which gets packages from the Python package index site. You can then install `requests` by running `pip install requests` at the command line.
- An access key (access key ID and secret access key) in environment variables named `AWS_ACCESS_KEY_ID` and `AWS_SECRET_ACCESS_KEY`. As a best practice, we recommend that you do not embed credentials in code. For more information, see Best Practices for Managing AWS Access Keys in the Amazon Web Services General Reference.

The following example shows how to make signed requests to Neptune using Python. The request makes a either a GET or POST request. Authentication information is passed using the `Authorization` request header.

**To make signed requests to the Gremlin and SPARQL Neptune endpoints**

1. Create a new file named `neptunesigv4.py`, and open it in a text editor.
2. Copy and paste the following code into the `neptunesigv4.py` file.

   ```python
   # Amazon Neptune version 4 signing example
   ```
# Connect to Neptune Using Python with Version 4 Signing (Example)

The following script requires python 3.6
# => the reason is that we're using urllib.parse() to manually encode URL
# parameters: the problem here is that SIGV4 encoding requires whitespaces
# to be encoded as %20 rather than not or using '+', as done by previous/
# default versions of the library.

# See: http://docs.aws.amazon.com/general/latest/gr/sigv4_signing.html
# This version makes a GET request and passes the signature
# in the Authorization header.
import http.client as http_client
import sys, os, base64, datetime, hashlib, hmac
import requests # pip install requests
import urllib
import os

# Configuration. http is required.
protocol = 'http'

# The following lines enable debugging at http/level (requests->urllib3->
# http.client)
# You will see the REQUEST, including HEADERS and DATA, and RESPONSE with HEADERS but
# without DATA.
#
# The only thing missing will be the response.body which is not logged.
#
# import logging
# http_client.HTTPConnection.debuglevel = 1
# logging.basicConfig()
# logging.getLogger().setLevel(logging.DEBUG)
# requests_log = logging.getLogger("requests.packages.urllib3")
# requests_log.setLevel(logging.DEBUG)
# requests_log.propagate = True

# Read AWS access key from env. variables. Best practice is NOT
# to embed credentials in code.
access_key = os.getenv('AWS_ACCESS_KEY_ID', '')
secret_key = os.getenv('AWS_SECRET_ACCESS_KEY', '')
region = os.getenv('REGION', '')

# make sure required params are passed
program_name = sys.argv[0]
if (len(sys.argv)!=5):
    print('"
    print('+++++ USAGE +++++')
    print("export AWS_ACCESS_KEY_ID=[MY_ACCESS_KEY_ID]"
    print("export AWS_SECRET_ACCESS_KEY=[MY_SECRET_ACCESS_KEY]"
    print("export REGION=[us-east-1|us-east-2|us-west-2|eu-west-1]"
    print("'+ program_name + ' [GET|POST] [host] [gremlin|sparql|sparqlupdate]
"[query_string]"
    print('"
    print('Examples:"
    print("python3.6 '+ program_name + ' my-neptune-instance:8182 GET sparql
"SELECT ?s WHERE { ?s ?p ?o }""
    print("python3.6 '+ program_name + ' my-neptune-instance:8182 POST sparql
"SELECT ?s WHERE { ?s ?p ?o }""
    print("python3.6 '+ program_name + ' my-neptune-instance:8182 POST sparqlupdate
    print("python3.6 '+ program_name + ' my-neptune-instance:8182 GET gremlin
"g.V().count()"
    print('"
    print('Environment variables must be defined as AWS_ACCESS_KEY_ID and
AWS_SECRET_ACCESS_KEY')
    print('"

API Version 2017-11-29
131
print('Current Limitations:)
print(' - Query mode "sparqlupdate" requires POST (as per the SPARQL 1.1 protocol)
print(' - The python script currently does not support POST for Gremlin')
print(')

if (access_key == ' '):
    print('!!! ERROR: Your AWS_ACCESS_KEY_ID environment variable is undefined.')
if (secret_key == ' '):
    print('!!! ERROR: Your AWS_SECRET_KEY environment variable is undefined.')
if (region == ' '):
    print('!!! ERROR: Your REGION environment variable is undefined.')
sys.exit()

# Read command line parameters
host = sys.argv[1]
method = sys.argv[2]
query_type = sys.argv[3]
query = sys.argv[4]

service = 'neptune-db'
endpoint = protocol + '://' + host

print()
print('+++++ USER INPUT +++++')
print('host = ' + host)
print('method = ' + method)
print('query_type = ' + query_type)
print('query = ' + query)

# Supporting GET and POST for now:
if (method != 'GET' and method != 'POST'):
    print('First parameter must be "GET" or "POST", but is "' + method + '".')
sys.exit()

# SPARQL UPDATE requires POST
if (method == 'GET' and query_type == 'sparqlupdate'):
    print('SPARQL UPDATE is not supported in GET mode. Please choose POST.')
sys.exit()

# Note: it looks like Gremlin POST requires the query to be encoded in a JSON struct; we haven't implemented this in the python script, so let's for now # disable Gremlin POST requests.
if (method == 'POST' and query_type == 'gremlin'):
    print('POST is currently not supported for Gremlin in this python script.')
sys.exit()

# Set the stack and payload depending on query_type.
if (query_type == 'sparql'):
    canonical_uri = '/sparql/
    payload = { 'query' : query }
elif (query_type == 'sparqlupdate'):
    canonical_uri = '/sparql/
    payload = { 'update' : query }
elif (query_type == 'gremlin'):
    canonical_uri = '/gremlin/
    payload = { 'gremlin' : query }
else:
Amazon Neptune User Guide  
Connect to Neptune Using Python  
with Version 4 Signing (Example)

print('Second parameter must be "gremlin" or "sparql", but is "' + method + "".')  
sys.exit()

# **************** REQUEST VALUES ****************

# do the encoding => quote_via=urllib.parse.quote is used to map " " => "%20"  
request_parameters = urllib.parse.urlencode(payload, quote_via=urllib.parse.quote)

# Key derivation functions. See:  
# http://docs.aws.amazon.com/general/latest/gr/signature-v4-examples.html#signature-v4-examples-python  
def sign(key, msg):  
    return hmac.new(key, msg.encode('utf-8'), hashlib.sha256).digest()

def getSignatureKey(key, dateStamp, regionName, serviceName):
    kDate = sign(('AWS4' + key).encode('utf-8'), dateStamp)
    kRegion = sign(kDate, regionName)
    kService = sign(kRegion, serviceName)
    kSigning = sign(kService, 'aws4_request')
    return kSigning

# **************** TASK 1: CREATE A CANONICAL REQUEST ****************
# http://docs.aws.amazon.com/general/latest/gr/sigv4-create-canonical-request.html  

# Step 1 is to define the verb (GET, POST, etc.)--already done.

# Create a date for headers and the credential string.
  t = datetime.datetime.utcnow()  
amzdate = t.strftime('%Y%m%dT%H%M%SZ')  
datestamp = t.strftime('%Y%m%d') # Date w/o time, used in credential scope

# **************** TASK 1: CREATE A CANONICAL REQUEST ****************
# http://docs.aws.amazon.com/general/latest/gr/sigv4-create-canonical-request.html  

# Step 1 is to define the verb (GET, POST, etc.)--already done.

# Step 2: is to define the canonical_uri--already done.

# Step 3: Create the canonical query string. In this example (a GET request),  
# request parameters are in the query string. Query string values must  
# be URL-encoded (space=%20). The parameters must be sorted by name.  
# For this example, the query string is pre-formatted in the request_parameters  
# variable.
  if (method == 'GET'):
    canonical_querystring = request_parameters
  elif (method == 'POST'):
    canonical_querystring = ''
  else:
    print('Request method is neither "GET" nor "POST", something is wrong here.')  
sys.exit()

# Step 4: Create the canonical headers and signed headers. Header names  
# must be trimmed and lowercase, and sorted in code point order from  
# low to high. Note that there is a trailing \n.
  canonical_headers = 'host:' + host + '\n' + 'x-amz-date:' + amzdate + '\n'

# Step 5: Create the list of signed headers. This lists the headers  
# in the canonical_headers list, delimited with ";" and in alpha order.  
# Note: The request can include any headers; canonical_headers and  
# signed_headers lists those that you want to be included in the  
# hash of the request. "Host" and "x-amz-date" are always required.
signed_headers = 'host;x-amz-date'

# Step 6: Create payload hash (hash of the request body content). For GET
# requests, the payload is an empty string ("").
if (method == 'GET'):
    post_payload = ''
elif (method == 'POST'):
    post_payload = request_parameters
else:
    print('Request method is neither "GET" nor "POST", something is wrong here.')
    sys.exit()

payload_hash = hashlib.sha256(post_payload.encode('utf-8')).hexdigest()

# Step 7: Combine elements to create canonical request.
canonical_request = method + '
' + canonical_uri + '
' + canonical_querystring + '
' + canonical_headers + '
' + signed_headers + '
' + payload_hash

# ************* TASK 2: CREATE THE STRING TO SIGN*************
# Match the algorithm to the hashing algorithm you use, either SHA-1 or
# SHA-256 (recommended)
algorithm = 'AWS4-HMAC-SHA256'
credential_scope = datestamp + '/' + region + '/' + service + '/' + 'aws4_request'
string_to_sign = algorithm + '
' + amzdate + '
' + credential_scope + '
' + hashlib.sha256(canonical_request.encode('utf-8')).hexdigest()

# ************* TASK 3: CALCULATE THE SIGNATURE *************
# Create the signing key using the function defined above.
signing_key = getSignatureKey(secret_key, datestamp, region, service)

# Sign the string_to_sign using the signing_key
signature = hmac.new(signKey, (string_to_sign).encode('utf-8')),
hashlib.sha256().hexdigest()

# ************* TASK 4: ADD SIGNING INFORMATION TO THE REQUEST *************
# The signing information can be either in a query string value or in
# a header named Authorization. This code shows how to use a header.
# Create authorization header and add to request headers
authorization_header = algorithm + ' ' + 'Credential=' + access_key + '/' +
credential_scope + ',', ' ' + 'SignedHeaders=' + signed_headers + ',', ' ' + 'Signature=' +
signature

# The request can include any headers, but MUST include "host", "x-amz-date",
# and (for this scenario) "Authorization". "host" and "x-amz-date" must
# be included in the canonical_headers and signed_headers, as noted
# earlier. Order here is not significant.
# Python note: The 'host' header is added automatically by the Python 'requests'
# library.
if (method == 'GET'):
    headers = {'x-amz-date':amzdate, 'Authorization':authorization_header}
elif (method == 'POST'):
    headers = {'content-type':'application/x-www-form-urlencoded', 'x-amz-
date':amzdate, 'Authorization':authorization_header}
else:
    print('Request method is neither "GET" nor "POST", something is wrong here.')
    sys.exit()

# ************* SEND THE REQUEST *************
request_url = endpoint + canonical_uri

print()
```python
if (method == 'GET'):
    print('++++ BEGIN GET REQUEST ++++')
    print('Request URL = ' + request_url)
    r = requests.get(request_url, headers=headers, verify=False, 
                     params=request_parameters)
    print()
    print('++++ RESPONSE ++++')
    print('Response code: %d
' % r.status_code)
    print(r.text)

elif (method == 'POST'):
    print('++++ BEGIN POST REQUEST ++++')
    print('Request URL = ' + request_url)
    r = requests.post(request_url, headers=headers, verify=False, 
                       data=request_parameters)
    print()
    print('++++ RESPONSE ++++')
    print('Response code: %d
' % r.status_code)
    print(r.text)
else:
    print('Request method is neither "GET" nor "POST", something is wrong here.')
sys.exit()
```

3. In a terminal, navigate to the location of the `neptunesigv4.py` file.
4. Type the following commands, replacing the access key, secret key, and region with the correct values.

```bash
export AWS_ACCESS_KEY_ID=MY_ACCESS_KEY_ID
export AWS_SECRET_ACCESS_KEY=MY_SECRET_ACCESS_KEY
export REGION=us-east-1|us-east-2|us-west-2|eu-west-1
```

5. Type one of the following commands to send a signed request to the Neptune DB instance. These examples use Python version 3.6.

**Gremlin**

```bash
python3.6 neptunesigv4.py your-neptune-endpoint GET gremlin "g.V().count()"
```

**SPARQL**

```bash
python3.6 neptunesigv4.py your-neptune-endpoint GET sparql "SELECT ?s WHERE { ?s ?p ?o }"
```

**SPARQL UPDATE**

```bash
python3.6 neptunesigv4.py your-neptune-endpoint POST sparqlupdate "INSERT DATA 
{ <http://s> <http://p> <http://o> }"
```

6. The syntax for running the Python script is as follows:

```bash
python neptunesigv4 [host]:[port] [GET|POST] [gremlin|sparql|sparqlupdate] 
"[query_string]"
```

Gremlin traversals only support GET. SPARQL UPDATE requires POST.
Appendix: Prerequisites on Amazon Linux EC2

The following are instructions for installing Apache Maven and Java 8 on an EC2 instance. These are required for the Neptune Sigv4 Authentication samples.

To Install Apache Maven and Java 8 on your EC2 instance

1. Connect to your Amazon EC2 instance with an SSH client.
2. Install Apache Maven on your EC2 instance. First, type the following to add a repository with a Maven package.

```bash
```

Type the following to set the version number for the packages.

```bash
sudo sed -i s/$releasever/6/g /etc/yum.repos.d/epel Apache-maven.repo
```

Then you can use `yum` to install Maven.

```bash
sudo yum install -y apache-maven
```

3. The Gremlin libraries require Java 8. Type the following to install Java 8 on your EC2 instance.

```bash
sudo yum install java-1.8.0-devel
```

4. Type the following to set Java 8 as the default runtime on your EC2 instance.

```bash
sudo /usr/sbin/alternatives --config java
```

When prompted, type the number for Java 8.

5. Type the following to set Java 8 as the default compiler on your EC2 instance.

```bash
sudo /usr/sbin/alternatives --config javac
```

When prompted, type the number for Java 8.
Manage Access with IAM

This section shows you how to select permissions that specify which Neptune actions a user or group can perform on which Neptune resources.

You can use AWS Identity and Access Management (IAM) and Neptune to help secure your resources by controlling who can access them.

**Note**  
If you want to use IAM to authenticate to a cluster or instance, see the IAM Database Authentication for Neptune (p. 122) section.

**Topics**
- Authentication (p. 137)
- Permissions Required to Use the Amazon Neptune Console (p. 138)
- AWS Managed (Predefined) Policies for Amazon Neptune (p. 138)
- Using Service-Linked Roles for Neptune (p. 139)

## Authentication

You can access AWS as any of the following types of identities:

- **AWS account root user** – When you first create an AWS account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account root user and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you do not use the root user for your everyday tasks, even the administrative ones. Instead, adhere to the best practice of using the root user only to create your first IAM user. Then securely lock away the root user credentials and use them to perform only a few account and service management tasks.

- **IAM user** – An IAM user is an identity within your AWS account that has specific custom permissions (for example, permissions to create a Neptune DB instance in Neptune). You can use an IAM user name and password to sign in to secure AWS webpages like the AWS Management Console, AWS Discussion Forums, or the AWS Support Center.

  In addition to a user name and password, you can also generate access keys for each user. You can use these keys when you access AWS services programmatically, either through one of the several SDKs or by using the AWS Command Line Interface (CLI). The SDK and CLI tools use the access keys to cryptographically sign your request. If you don’t use AWS tools, you must sign the request yourself. Neptune supports Signature Version 4, a protocol for authenticating inbound API requests. For more information about authenticating requests, see Signature Version 4 Signing Process in the AWS General Reference.

- **IAM role** – An IAM role is an IAM identity that you can create in your account that has specific permissions. It is similar to an IAM user, but it is not associated with a specific person. An IAM role enables you to obtain temporary access keys that can be used to access AWS services and resources. IAM roles with temporary credentials are useful in the following situations:

  - **Federated user access** – Instead of creating an IAM user, you can use existing user identities from AWS Directory Service, your enterprise user directory, or a web identity provider. These are known as federated users. AWS assigns a role to a federated user when access is requested through an identity provider. For more information about federated users, see Federated Users and Roles in the IAM User Guide.
• **AWS service access** – You can use an IAM role in your account to grant an AWS service permissions to access your account’s resources. For example, you can create a role that allows Amazon Redshift to access an Amazon S3 bucket on your behalf and then load data from that bucket into an Amazon Redshift cluster. For more information, see [Creating a Role to Delegate Permissions to an AWS Service](https://docs.aws.amazon.com/IAM/latest/UserGuide/id_use-service-role-nonhosted-account.html) in the IAM User Guide.

• **Applications running on Amazon EC2** – You can use an IAM role to manage temporary credentials for applications that are running on an EC2 instance and making AWS API requests. This is preferable to storing access keys within the EC2 instance. To assign an AWS role to an EC2 instance and make it available to all of its applications, you create an instance profile that is attached to the instance. An instance profile contains the role and enables programs that are running on the EC2 instance to get temporary credentials. For more information, see [Using an IAM Role to Grant Permissions to Applications Running on Amazon EC2 Instances](https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_ec2.html) in the IAM User Guide.

### Permissions Required to Use the Amazon Neptune Console

For a user to work with the Amazon Neptune console, that user must have a minimum set of permissions. These permissions allow the user to describe the Neptune resources for their AWS account and to provide other related information, including Amazon EC2 security and network information.

If you create an IAM policy that is more restrictive than the minimum required permissions, the console won’t function as intended for users with that IAM policy. To ensure that those users can still use the Neptune console, also attach the NeptuneReadOnlyAccess managed policy to the user, as described in [AWS Managed (Predefined) Policies for Amazon Neptune](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html#access_policies_managed) (p. 138).

You don’t need to allow minimum console permissions for users that are making calls only to the AWS CLI or the Amazon Neptune API.

### AWS Managed (Predefined) Policies for Amazon Neptune

AWS addresses many common use cases by providing standalone IAM policies that are created and administered by AWS. Managed policies grant necessary permissions for common use cases so you can avoid having to investigate what permissions are needed. For more information, see [AWS Managed Policies](https://docs.aws.amazon.com/IAM/latest/UserGuide/managed-policies.html) in the IAM User Guide.

The following AWS managed policies, which you can attach to users in your account, are specific to Amazon Neptune:

• **NeptuneReadOnlyAccess** – Grants read-only access to all Amazon Neptune resources for the root AWS account.

• **NeptuneFullAccess** – Grants full access to all Amazon Neptune resources for the root AWS account.

**Important**

These IAM roles grant some access to RDS resources. For certain management features Neptune uses operational technology that is shared with Amazon RDS.

You can also create custom IAM policies that allow users to access the required Amazon Neptune API actions and resources. You can attach these custom policies to the IAM users or groups that require those permissions.
For certain management features Amazon Neptune uses operational technology that is shared with Amazon RDS. This includes management API permissions. To restrict access to a specific set of actions, see Access Control Overview in the Amazon RDS documentation.

### Using Service-Linked Roles for Neptune

Amazon Neptune uses AWS Identity and Access Management (IAM) service-linked roles. A service-linked role is a unique type of IAM role that is linked directly to Neptune. Service-linked roles are predefined by Neptune and include all the permissions that the service requires to call other AWS services on your behalf.

**Important**

For certain management features Amazon Neptune uses operational technology that is shared with Amazon RDS. This includes the Service-Linked Role and management API permissions.

A service-linked role makes using Neptune easier because you don’t have to manually add the necessary permissions. Neptune defines the permissions of its service-linked roles, and unless defined otherwise, only Neptune can assume its roles. The defined permissions include the trust policy and the permissions policy, and that permissions policy cannot be attached to any other IAM entity.

You can delete the roles only after first deleting their related resources. This protects your Neptune resources because you can’t inadvertently remove permission to access the resources.

For information about other services that support service-linked roles, see AWS Services That Work with IAM and look for the services that have Yes in the Service-Linked Role column. Choose a Yes with a link to view the service-linked role documentation for that service.

### Service-Linked Role Permissions for Neptune

Neptune uses the service-linked role named AWSServiceRoleForRDS – to allow Neptune and Amazon RDS to call AWS services on behalf of your database instances.

**Important**

For certain management features Amazon Neptune uses operational technology that is shared with Amazon RDS. This includes the Service-Linked Role and management API permissions.

The AWSServiceRoleForRDS service-linked role trusts the following services to assume the role:

- rds.amazonaws.com

The role permissions policy allows Neptune to complete the following actions on the specified resources:

- Actions on ec2:
  - AssignPrivateIpAddresses
  - AuthorizeSecurityGroupIngress
  - CreateNetworkInterface
  - CreateSecurityGroup
  - DeleteNetworkInterface
  - DeleteSecurityGroup
  - DescribeAvailabilityZones
  - DescribeInternetGateways
  - DescribeSecurityGroups
  - DescribeSubnets
  - DescribeVpcAttribute

API Version 2017-11-29
Creating a Service-Linked Role for Neptune

You don’t need to manually create a service-linked role. When you create an instance or a cluster, Neptune creates the service-linked role for you.

Important
To learn more, see A New Role Appeared in My IAM Account.

If you delete this service-linked role, and then need to create it again, you can use the same process to recreate the role in your account. When you create an instance or a cluster, Neptune creates the service-linked role for you again.

Editing a Service-Linked Role for Neptune

Neptune does not allow you to edit the AWSServiceRoleForRDS service-linked role. After you create a service-linked role, you cannot change the name of the role because various entities might reference the...
Deleting a Service-Linked Role for Neptune

If you no longer need to use a feature or service that requires a service-linked role, we recommend that you delete that role. That way you don’t have an unused entity that is not actively monitored or maintained. However, you must delete all of your instances and clusters before you can delete the service-linked role.

Cleaning Up a Service-Linked Role

Before you can use IAM to delete a service-linked role, you must first confirm that the role has no active sessions and remove any resources used by the role.

To check whether the service-linked role has an active session in the IAM console

1. Sign in to the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane of the IAM console, choose Roles. Then choose the name (not the check box) of the AWSServiceRoleForRDS role.
3. On the Summary page for the selected role, choose the Access Advisor tab.
4. On the Access Advisor tab, review recent activity for the service-linked role.

   **Note**
   If you are unsure whether Neptune is using the AWSServiceRoleForRDS role, you can try to delete the role. If the service is using the role, then the deletion fails and you can view the regions where the role is being used. If the role is being used, then you must wait for the session to end before you can delete the role. You cannot revoke the session for a service-linked role.

If you want to remove the AWSServiceRoleForRDS role, you must first delete all of your instances and clusters.

Deleting All of Your Instances

Use one of these procedures to delete each of your instances.

**To delete an instance (console)**

1. Open the Amazon RDS console at https://console.aws.amazon.com/rds/.
2. In the navigation pane, choose Instances.
3. In the Instances list, choose the instance that you want to delete.
4. Choose Instance actions, and then choose Delete.
5. If you are prompted for Create final Snapshot?, choose Yes or No.
6. If you chose Yes in the previous step, for Final snapshot name type the name of your final snapshot.
7. Choose Delete.

**To delete an instance (CLI)**

See delete-db-instance in the AWS CLI Command Reference.

**To delete an instance (API)**
Deleting All of Your Clusters

Use one of the following procedures to delete a single cluster. Repeat the procedure for each of your clusters.

To delete a cluster (console)

1. Sign in to the AWS Management Console, and open the Amazon Neptune console at https://console.aws.amazon.com/neptune/home.
2. In the Clusters list, choose the cluster that you want to delete.
3. Choose Cluster Actions, and then choose Delete.
4. Choose Delete.

To delete a cluster (CLI)

See delete-db-cluster in the AWS CLI Command Reference.

To delete a cluster (API)

See DeleteDBCluster

You can use the IAM console, the IAM CLI, or the IAM API to delete the AWSServiceRoleForRDS service-linked role. For more information, see Deleting a Service-Linked Role in the IAM User Guide.
Amazon Neptune Limits

Regions
Amazon Neptune is available in the following regions:

- US East (N. Virginia)
- US East (Ohio)
- US West (Oregon)
- EU (Ireland)

Instance Sizes
Neptune supports the `db.r4.large`, `db.r4.xlarge`, `db.r4.2xlarge`, `db.r4.4xlarge`, and `db.r4.8xlarge` DB instance classes.

Instance Limit
Amazon Neptune shares a DB instance limit with Amazon RDS.

For certain management features Neptune uses operational technology that is shared with Amazon RDS. The default limit for Neptune and Amazon RDS instances is 40.

You can request an increase on this limit. For more information, see https://aws.amazon.com/support.

Account Limits
The following are per-account limits.

<table>
<thead>
<tr>
<th>Item</th>
<th>Limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clusters</td>
<td>20</td>
</tr>
<tr>
<td>DB Subnet Groups</td>
<td>50</td>
</tr>
<tr>
<td>DB Snapshots</td>
<td>100</td>
</tr>
<tr>
<td>DB Security Groups (Per VPC)</td>
<td>25</td>
</tr>
</tbody>
</table>

You can request an increase on some limits. For more information, see https://aws.amazon.com/support.

Console Access Required
Creating and modifying Amazon Neptune instances and clusters must be done through the AWS Management Console.

VPC Required
Amazon Neptune is a virtual private cloud (VPC)–only service. Additionally, instances do not allow access from outside the VPC.

Availability Zones and DB Subnet Groups
Amazon Neptune requires a DB subnet group for each cluster that has subnets in at least two supported Availability Zones. We recommend using three or more subnets in different Availability Zones.
Gremlin Implementation

The Amazon Neptune Gremlin implementation has specific implementation details. For more information, see Neptune Gremlin Implementation Differences (p. 27).

SPARQL UPDATE LOAD

SPARQL UPDATE LOAD from URI works only with resources that are within the same VPC. This includes Amazon S3 URLs in the same region as the cluster with an Amazon S3 VPC endpoint created. For information about creating a VPC endpoint, see Amazon S3 VPC Endpoint (p. 68).

The Amazon S3 URL must be HTTPS, and any authentication must be included in the URL. For more information, see Authenticating Requests: Using Query Parameters.

If you need to load data from a file, we recommend that you use the Amazon Neptune loader API. For more information, see Loading Data into Neptune (p. 59).

Note
The Amazon Neptune loader API is non-ACID.

Authentication and Access

IAM authentication and access control are not supported for Gremlin, SPARQL. It is only supported at the DB Cluster level. See IAM Database Authentication for Neptune (p. 122)

The Amazon Neptune console requires NeptuneReadOnlyAccess permissions. You can restrict access to IAM users by revoking this access. For more information, see AWS Managed (Predefined) Policies for Amazon Neptune (p. 138)

Amazon Neptune does not support user name/password–based access control.
The following table describes important changes to the documentation for Amazon Neptune.

- **Latest API version:** 2018-05-30
- **Latest documentation update:** May 30, 2018

<table>
<thead>
<tr>
<th>Change</th>
<th>Description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon Neptune initial release</td>
<td>This is the initial release of the <em>Neptune User Guide.</em></td>
<td>May 30, 2018</td>
</tr>
</tbody>
</table>
Neptune Cluster, Instance, and Resource Management API Reference

This section describes the cluster, instance, and resource management operations for Amazon Neptune DB instances that are accessible via HTTP, the command line interface (AWS CLI), or the AWS SDK. You can use these APIs to create, delete, and modify clusters and instances.

**Important**
These APIs are only used for managing clusters, instances, and related resources. For information about how to connect to a running Neptune DB cluster, see Accessing a Neptune Graph (p. 24).

Topics
- Actions (p. 146)
- Data Types (p. 299)
- Common Errors (p. 363)
- Common Parameters (p. 365)

Actions

The following actions are supported:

- AddRoleToDBCluster (p. 148)
- AddSourceIdentifierToSubscription (p. 150)
- AddTagsToResource (p. 152)
- ApplyPendingMaintenanceAction (p. 154)
- CopyDBClusterParameterGroup (p. 156)
- CopyDBClusterSnapshot (p. 158)
- CopyDBParameterGroup (p. 162)
- CreateDBCluster (p. 164)
- CreateDBClusterParameterGroup (p. 171)
- CreateDBClusterSnapshot (p. 173)
- CreateDBInstance (p. 175)
- CreateDBParameterGroup (p. 185)
- CreateDBSubnetGroup (p. 187)
- CreateEventSubscription (p. 189)
- DeleteDBCluster (p. 192)
- DeleteDBClusterParameterGroup (p. 194)
- DeleteDBClusterSnapshot (p. 196)
- DeleteDBInstance (p. 198)
- DeleteDBParameterGroup (p. 201)
• DeleteDBSubnetGroup (p. 203)
• DeleteEventSubscription (p. 205)
• DescribeDBClusterParameterGroups (p. 206)
• DescribeDBClusterParameters (p. 208)
• DescribeDBClusters (p. 210)
• DescribeDBClusterSnapshots (p. 214)
• DescribeDBEngineVersions (p. 217)
• DescribeDBInstances (p. 220)
• DescribeDBParameterGroups (p. 222)
• DescribeDBParameters (p. 224)
• DescribeDBSubnetGroups (p. 226)
• DescribeEngineDefaultClusterParameters (p. 228)
• DescribeEngineDefaultParameters (p. 230)
• DescribeEventCategories (p. 232)
• DescribeEvents (p. 234)
• DescribeEventSubscriptions (p. 237)
• DescribeOrderableDBInstanceOptions (p. 239)
• DescribePendingMaintenanceActions (p. 242)
• DescribeValidDBInstanceModifications (p. 244)
• FailoverDBCluster (p. 246)
• ListTagsForResource (p. 248)
• ModifyDBCluster (p. 250)
• ModifyDBClusterParameterGroup (p. 255)
• ModifyDBClusterSnapshotAttribute (p. 257)
• ModifyDBInstance (p. 260)
• ModifyDBParameterGroup (p. 270)
• ModifyDBSubnetGroup (p. 272)
• ModifyEventSubscription (p. 274)
• PromoteReadReplicaDBCluster (p. 276)
• RebootDBInstance (p. 278)
• RemoveRoleFromDBCluster (p. 280)
• RemoveSourceIdentifierFromSubscription (p. 282)
• RemoveTagsFromResource (p. 284)
• ResetDBClusterParameterGroup (p. 286)
• ResetDBParameterGroup (p. 288)
• RestoreDBClusterFromSnapshot (p. 290)
• RestoreDBClusterToPointInTime (p. 295)
AddRoleToDBCluster

Associates an Identity and Access Management (IAM) role from a Neptune DB cluster.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

DBClusterIdentifier

The name of the DB cluster to associate the IAM role with.

Type: String

Required: Yes

RoleArn

The Amazon Resource Name (ARN) of the IAM role to associate with the Neptune DB cluster, for example `arn:aws:iam::123456789012:role/NeptuneAccessRole`.

Type: String

Required: Yes

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

DBClusterNotFoundFault

`DBClusterIdentifier` does not refer to an existing DB cluster.

HTTP Status Code: 404

DBClusterRoleAlreadyExists

The specified IAM role Amazon Resource Name (ARN) is already associated with the specified DB cluster.

HTTP Status Code: 400

DBClusterRoleQuotaExceeded

You have exceeded the maximum number of IAM roles that can be associated with the specified DB cluster.

HTTP Status Code: 400

InvalidDBClusterStateFault

The DB cluster is not in a valid state.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:
• AWS Command Line Interface
• AWS SDK for .NET
• AWS SDK for C++
• AWS SDK for Go
• AWS SDK for Java
• AWS SDK for JavaScript
• AWS SDK for PHP V3
• AWS SDK for Python
• AWS SDK for Ruby V2
AddSourceIdentifierToSubscription

Adds a source identifier to an existing event notification subscription.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

SourceIdentifier

The identifier of the event source to be added.

Constraints:
- If the source type is a DB instance, then a DBInstanceIdentifier must be supplied.
- If the source type is a DB security group, a DBSecurityGroupName must be supplied.
- If the source type is a DB parameter group, a DBParameterGroupName must be supplied.
- If the source type is a DB snapshot, a DBSnapshotIdentifier must be supplied.

Type: String
Required: Yes

SubscriptionName

The name of the event notification subscription you want to add a source identifier to.

Type: String
Required: Yes

Response Elements

The following element is returned by the service.

EventSubscription

Contains the results of a successful invocation of the DescribeEventSubscriptions (p. 237) action.

Type: EventSubscription (p. 340) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

SourceNotFound

HTTP Status Code: 404

SubscriptionNotFoundError

HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:
• AWS Command Line Interface
• AWS SDK for .NET
• AWS SDK for C++
• AWS SDK for Go
• AWS SDK for Java
• AWS SDK for JavaScript
• AWS SDK for PHP V3
• AWS SDK for Python
• AWS SDK for Ruby V2
AddTagsToResource

Adds metadata tags to an Amazon Neptune resource. These tags can also be used with cost allocation reporting to track cost associated with Amazon Neptune resources, or used in a Condition statement in an IAM policy for Amazon Neptune.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

ResourceName

The Amazon Neptune resource that the tags are added to. This value is an Amazon Resource Name (ARN). For information about creating an ARN, see Constructing an Amazon Resource Name (ARN).

Type: String

Required: Yes

Tags.Tag.N

The tags to be assigned to the Amazon Neptune resource.

Type: Array of Tag (p. 358) objects

Required: Yes

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

DBClusterNotFoundFault

*DBClusterIdentifier* does not refer to an existing DB cluster.

HTTP Status Code: 404

DBInstanceNotFound

*DBInstanceIdentifier* does not refer to an existing DB instance.

HTTP Status Code: 404

DBSnapshotNotFound

*DBSnapshotIdentifier* does not refer to an existing DB snapshot.

HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
• AWS SDK for Java
• AWS SDK for JavaScript
• AWS SDK for PHP V3
• AWS SDK for Python
• AWS SDK for Ruby V2
ApplyPendingMaintenanceAction

Applies a pending maintenance action to a resource (for example, to a DB instance).

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

ApplyAction

The pending maintenance action to apply to this resource.

Valid values: system-update, db-upgrade

Type: String

Required: Yes

OptInType

A value that specifies the type of opt-in request, or undoes an opt-in request. An opt-in request of type immediate can't be undone.

Valid values:
- immediate - Apply the maintenance action immediately.
- next-maintenance - Apply the maintenance action during the next maintenance window for the resource.
- undo-opt-in - Cancel any existing next-maintenance opt-in requests.

Type: String

Required: Yes

ResourceIdentifier

The Amazon Resource Name (ARN) of the resource that the pending maintenance action applies to. For information about creating an ARN, see Constructing an Amazon Resource Name (ARN).

Type: String

Required: Yes

Response Elements

The following element is returned by the service.

ResourcePendingMaintenanceActions

Describes the pending maintenance actions for a resource.

Type: ResourcePendingMaintenanceActions (p. 356) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).
ResourceNotFoundException

The specified resource ID was not found.

HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
CopyDBClusterParameterGroup

Copies the specified DB cluster parameter group.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

SourceDBClusterParameterGroupIdentifier

The identifier or Amazon Resource Name (ARN) for the source DB cluster parameter group. For information about creating an ARN, see Constructing an Amazon Resource Name (ARN).

Constraints:
- Must specify a valid DB cluster parameter group.
- If the source DB cluster parameter group is in the same AWS Region as the copy, specify a valid DB parameter group identifier, for example my-db-cluster-param-group, or a valid ARN.
- If the source DB parameter group is in a different AWS Region than the copy, specify a valid DB cluster parameter group ARN, for example arn:aws:rds:us-east-1:123456789012:cluster-pg:custom-cluster-group1.

Type: String

Required: Yes

Tags.Tag.N

A list of tags. For more information, see Tagging Amazon Neptune Resources.

Type: Array of Tag (p. 358) objects

Required: No

TargetDBClusterParameterGroupDescription

A description for the copied DB cluster parameter group.

Type: String

Required: Yes

TargetDBClusterParameterGroupIdentifier

The identifier for the copied DB cluster parameter group.

Constraints:
- Cannot be null, empty, or blank
- Must contain from 1 to 255 letters, numbers, or hyphens
- First character must be a letter
- Cannot end with a hyphen or contain two consecutive hyphens

Example: my-cluster-param-group1

Type: String

Required: Yes

API Version 2017-11-29

156
Response Elements

The following element is returned by the service.

**DBClusterParameterGroup**

Contains the details of an Amazon Neptune DB cluster parameter group.

This data type is used as a response element in the **DescribeDBClusterParameterGroups** (p. 206) action.

Type: **DBClusterParameterGroup** (p. 311) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

**DBParameterGroupAlreadyExists**

A DB parameter group with the same name exists.

HTTP Status Code: 400

**DBParameterGroupName** does not refer to an existing DB parameter group.

HTTP Status Code: 404

**DBParameterGroupQuotaExceeded**

Request would result in user exceeding the allowed number of DB parameter groups.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
CopyDBClusterSnapshot

Copies a snapshot of a DB cluster.

To copy a DB cluster snapshot from a shared manual DB cluster snapshot, `SourceDBClusterSnapshotIdentifier` must be the Amazon Resource Name (ARN) of the shared DB cluster snapshot.

You can copy an encrypted DB cluster snapshot from another AWS Region. In that case, the AWS Region where you call the `CopyDBClusterSnapshot` action is the destination AWS Region for the encrypted DB cluster snapshot to be copied to. To copy an encrypted DB cluster snapshot from another AWS Region, you must provide the following values:

- **KmsKeyId** - The AWS Key Management System (AWS KMS) key identifier for the key to use to encrypt the copy of the DB cluster snapshot in the destination AWS Region.
- **PreSignedUrl** - A URL that contains a Signature Version 4 signed request for the `CopyDBClusterSnapshot` action to be called in the source AWS Region where the DB cluster snapshot is copied from. The pre-signed URL must be a valid request for the `CopyDBClusterSnapshot` API action that can be executed in the source AWS Region that contains the encrypted DB cluster snapshot to be copied.

The pre-signed URL request must contain the following parameter values:

- **KmsKeyId** - The KMS key identifier for the key to use to encrypt the copy of the DB cluster snapshot in the destination AWS Region. This is the same identifier for both the `CopyDBClusterSnapshot` action that is called in the destination AWS Region, and the action contained in the pre-signed URL.
- **DestinationRegion** - The name of the AWS Region that the DB cluster snapshot will be created in.
- **SourceDBClusterSnapshotIdentifier** - The DB cluster snapshot identifier for the encrypted DB cluster snapshot to be copied. This identifier must be in the Amazon Resource Name (ARN) format for the source AWS Region. For example, if you are copying an encrypted DB cluster snapshot from the us-west-2 AWS Region, then your `SourceDBClusterSnapshotIdentifier` looks like the following example: `arn:aws:rds:us-west-2:123456789012:cluster-snapshot:neptune-cluster1-snapshot-20161115`.

To learn how to generate a Signature Version 4 signed request, see [Authenticating Requests: Using Query Parameters (AWS Signature Version 4)](https://docs.aws.amazon.com/general/latest/gr/signature-v4-example.html) and [Signature Version 4 Signing Process](https://docs.aws.amazon.com/general/latest/gr/signature-v4-example.html).

- **TargetDBClusterSnapshotIdentifier** - The identifier for the new copy of the DB cluster snapshot in the destination AWS Region.
- **SourceDBClusterSnapshotIdentifier** - The DB cluster snapshot identifier for the encrypted DB cluster snapshot to be copied. This identifier must be in the ARN format for the source AWS Region and is the same value as the `SourceDBClusterSnapshotIdentifier` in the pre-signed URL.

To cancel the copy operation once it is in progress, delete the target DB cluster snapshot identified by `TargetDBClusterSnapshotIdentifier` while that DB cluster snapshot is in "copying" status.

**Request Parameters**

For information about the parameters that are common to all actions, see [Common Parameters (p. 365)](#).

**CopyTags**

True to copy all tags from the source DB cluster snapshot to the target DB cluster snapshot, and otherwise false. The default is false.

Type: Boolean
KmsKeyId

The AWS AWS KMS key ID for an encrypted DB cluster snapshot. The KMS key ID is the Amazon Resource Name (ARN), KMS key identifier, or the KMS key alias for the KMS encryption key.

If you copy an unencrypted DB cluster snapshot and specify a value for the KmsKeyId parameter, Amazon Neptune encrypts the target DB cluster snapshot using the specified KMS encryption key.

If you copy an encrypted DB cluster snapshot from your AWS account, you can specify a value for KmsKeyId to encrypt the copy with a new KMS encryption key. If you don't specify a value for KmsKeyId, then the copy of the DB cluster snapshot is encrypted with the same KMS key as the source DB cluster snapshot.

If you copy an encrypted DB cluster snapshot that is shared from another AWS account, then you must specify a value for KmsKeyId.

To copy an encrypted DB cluster snapshot to another AWS Region, you must set KmsKeyId to the KMS key ID you want to use to encrypt the copy of the DB cluster snapshot in the destination AWS Region. KMS encryption keys are specific to the AWS Region that they are created in, and you can't use encryption keys from one AWS Region in another AWS Region.

Type: String

PreSignedUrl

The URL that contains a Signature Version 4 signed request for the CopyDBClusterSnapshot API action in the AWS Region that contains the source DB cluster snapshot to copy. The PreSignedUrl parameter must be used when copying an encrypted DB cluster snapshot from another AWS Region.

The pre-signed URL must be a valid request for the CopyDBClusterSnapshot API action that can be executed in the source AWS Region that contains the encrypted DB cluster snapshot to be copied. The pre-signed URL request must contain the following parameter values:

- KmsKeyId - The AWS KMS key identifier for the key to use to encrypt the copy of the DB cluster snapshot in the destination AWS Region. This is the same identifier for both the CopyDBClusterSnapshot action that is called in the destination AWS Region, and the action contained in the pre-signed URL.
- DestinationRegion - The name of the AWS Region that the DB cluster snapshot will be created in.
- SourceDBClusterSnapshotIdentifier - The DB cluster snapshot identifier for the encrypted DB cluster snapshot to be copied. This identifier must be in the Amazon Resource Name (ARN) format for the source AWS Region. For example, if you are copying an encrypted DB cluster snapshot from the us-west-2 AWS Region, then your SourceDBClusterSnapshotIdentifier looks like the following example: arn:aws:rds:us-west-2:123456789012:cluster-neptune-cluster1-snapshot-20161115.

To learn how to generate a Signature Version 4 signed request, see Authenticating Requests: Using Query Parameters (AWS Signature Version 4) and Signature Version 4 Signing Process.

Type: String

SourceDBClusterSnapshotIdentifier

The identifier of the DB cluster snapshot to copy. This parameter is not case-sensitive.

You can’t copy an encrypted, shared DB cluster snapshot from one AWS Region to another.
CopyDBClusterSnapshot

Constraints:
- Must specify a valid system snapshot in the "available" state.
- If the source snapshot is in the same AWS Region as the copy, specify a valid DB snapshot identifier.
- If the source snapshot is in a different AWS Region than the copy, specify a valid DB cluster snapshot ARN.

Example: my-cluster-snapshot1

Type: String
Required: Yes

Tags.Tag.N

A list of tags. For more information, see Tagging Amazon Neptune Resources.

Type: Array of Tag (p. 358) objects
Required: No

TargetDBClusterSnapshotIdentifier

The identifier of the new DB cluster snapshot to create from the source DB cluster snapshot. This parameter is not case-sensitive.

Constraints:
- Must contain from 1 to 63 letters, numbers, or hyphens.
- First character must be a letter.
- Cannot end with a hyphen or contain two consecutive hyphens.

Example: my-cluster-snapshot2

Type: String
Required: Yes

Response Elements

The following element is returned by the service.

DBClusterSnapshot

Contains the details for an Amazon Neptune DB cluster snapshot

This data type is used as a response element in the DescribeDBClusterSnapshots (p. 214) action.

Type: DBClusterSnapshot (p. 313) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

DBClusterSnapshotAlreadyExistsFault

User already has a DB cluster snapshot with the given identifier.

HTTP Status Code: 400
DBClusterSnapshotNotFoundFault

*DBClusterSnapshotIdentifier* does not refer to an existing DB cluster snapshot.

HTTP Status Code: 404

InvalidDBClusterSnapshotStateFault

The supplied value is not a valid DB cluster snapshot state.

HTTP Status Code: 400

InvalidDBClusterStateFault

The DB cluster is not in a valid state.

HTTP Status Code: 400

KMSKeyNotAccessibleFault

Error accessing KMS key.

HTTP Status Code: 400

SnapshotQuotaExceeded

Request would result in user exceeding the allowed number of DB snapshots.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
CopyDBParameterGroup

Copies the specified DB parameter group.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

SourceDBParameterGroupIdentifier

The identifier or ARN for the source DB parameter group. For information about creating an ARN, see Constructing an Amazon Resource Name (ARN).

Constraints:
- Must specify a valid DB parameter group.
- Must specify a valid DB parameter group identifier, for example my-db-param-group, or a valid ARN.

Type: String
Required: Yes

Tags.Tag.N

A list of tags. For more information, see Tagging Amazon Neptune Resources.

Type: Array of Tag (p. 358) objects
Required: No

TargetDBParameterGroupDescription

A description for the copied DB parameter group.

Type: String
Required: Yes

TargetDBParameterGroupIdentifier

The identifier for the copied DB parameter group.

Constraints:
- Cannot be null, empty, or blank
- Must contain from 1 to 255 letters, numbers, or hyphens
- First character must be a letter
- Cannot end with a hyphen or contain two consecutive hyphens

Example: my-db-parameter-group

Type: String
Required: Yes

Response Elements

The following element is returned by the service.
DBParameterGroup

Contains the details of an Amazon Neptune DB parameter group.

This data type is used as a response element in the DescribeDBParameterGroups (p. 222) action.

Type: DBParameterGroup (p. 328) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

DBParameterGroupAlreadyExists

A DB parameter group with the same name exists.

HTTP Status Code: 400

DBParameterGroupNotFound

DBParameterGroupName does not refer to an existing DB parameter group.

HTTP Status Code: 404

DBParameterGroupQuotaExceeded

Request would result in user exceeding the allowed number of DB parameter groups.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
CreateDBCluster

Creates a new Amazon Neptune DB cluster.

You can use the `ReplicationSourceIdentifier` parameter to create the DB cluster as a Read Replica of another DB cluster or Amazon Neptune DB instance. For cross-region replication where the DB cluster identified by `ReplicationSourceIdentifier` is encrypted, you must also specify the `PreSignedUrl` parameter.

**Request Parameters**

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

**AvailabilityZones.AvailabilityZone.N**

A list of EC2 Availability Zones that instances in the DB cluster can be created in.

Type: Array of strings

Required: No

**BackupRetentionPeriod**

The number of days for which automated backups are retained. You must specify a minimum value of 1.

Default: 1

Constraints:
- Must be a value from 1 to 35

Type: Integer

Required: No

**CharacterSetName**

A value that indicates that the DB cluster should be associated with the specified CharacterSet.

Type: String

Required: No

**DatabaseName**

The name for your database of up to 64 alpha-numeric characters. If you do not provide a name, Amazon Neptune will not create a database in the DB cluster you are creating.

Type: String

Required: No

**DBClusterIdentifier**

The DB cluster identifier. This parameter is stored as a lowercase string.

Constraints:
- Must contain from 1 to 63 letters, numbers, or hyphens.
- First character must be a letter.
- Cannot end with a hyphen or contain two consecutive hyphens.
Example: my-cluster1

Type: String

Required: Yes

**DBClusterParameterGroupName**

The name of the DB cluster parameter group to associate with this DB cluster. If this argument is omitted, the default is used.

Constraints:
- If supplied, must match the name of an existing DBClusterParameterGroup.

Type: String

Required: No

**DBSubnetGroupName**

A DB subnet group to associate with this DB cluster.

Constraints: Must match the name of an existing DBSubnetGroup. Must not be default.

Example: mySubnetgroup

Type: String

Required: No

**EnableIAMDatabaseAuthentication**

True to enable mapping of AWS Identity and Access Management (IAM) accounts to database accounts, and otherwise false.

Default: false

Type: Boolean

Required: No

**Engine**

The name of the database engine to be used for this DB cluster.

Valid Values: neptune

Type: String

Required: Yes

**EngineVersion**

The version number of the database engine to use.

Example: 1.0.1

Type: String

Required: No

**KmsKeyId**

The AWS KMS key identifier for an encrypted DB cluster.

The KMS key identifier is the Amazon Resource Name (ARN) for the KMS encryption key. If you are creating a DB cluster with the same AWS account that owns the KMS encryption key used to encrypt
the new DB cluster, then you can use the KMS key alias instead of the ARN for the KMS encryption key.

If an encryption key is not specified in KmsKeyId:
• If ReplicationSourceIdentifier identifies an encrypted source, then Amazon Neptune will use the encryption key used to encrypt the source. Otherwise, Amazon Neptune will use your default encryption key.
• If the StorageEncrypted parameter is true and ReplicationSourceIdentifier is not specified, then Amazon Neptune will use your default encryption key.

AWS KMS creates the default encryption key for your AWS account. Your AWS account has a different default encryption key for each AWS Region.

If you create a Read Replica of an encrypted DB cluster in another AWS Region, you must set KmsKeyId to a KMS key ID that is valid in the destination AWS Region. This key is used to encrypt the Read Replica in that AWS Region.

Type: String
Required: No

MasterUsername
The name of the master user for the DB cluster.

Constraints:
• Must be 1 to 16 letters or numbers.
• First character must be a letter.
• Cannot be a reserved word for the chosen database engine.

Type: String
Required: No

MasterUserPassword
The password for the master database user. This password can contain any printable ASCII character except "/", ",", or "."

Constraints: Must contain from 8 to 41 characters.

Type: String
Required: No

OptionGroupName
A value that indicates that the DB cluster should be associated with the specified option group.

Permanent options can't be removed from an option group. The option group can't be removed from a DB cluster once it is associated with a DB cluster.

Type: String
Required: No

Port
The port number on which the instances in the DB cluster accept connections.

Default: 8182

Type: Integer
Required: No

**PreferredBackupWindow**

The daily time range during which automated backups are created if automated backups are enabled using the `BackupRetentionPeriod` parameter.

The default is a 30-minute window selected at random from an 8-hour block of time for each AWS Region. To see the time blocks available, see Adjusting the Preferred Maintenance Window in the Amazon Neptune User Guide.

Constraints:
- Must be in the format `hh24:mi-hh24:mi`.
- Must be in Universal Coordinated Time (UTC).
- Must not conflict with the preferred maintenance window.
- Must be at least 30 minutes.

Type: String

Required: No

**PreferredMaintenanceWindow**

The weekly time range during which system maintenance can occur, in Universal Coordinated Time (UTC).


The default is a 30-minute window selected at random from an 8-hour block of time for each AWS Region, occurring on a random day of the week. To see the time blocks available, see Adjusting the Preferred Maintenance Window in the Amazon Neptune User Guide.

Valid Days: Mon, Tue, Wed, Thu, Fri, Sat, Sun.

Constraints: Minimum 30-minute window.

Type: String

Required: No

**PreSignedUrl**

A URL that contains a Signature Version 4 signed request for the `CreateDBCluster` action to be called in the source AWS Region where the DB cluster is replicated from. You only need to specify `PreSignedUrl` when you are performing cross-region replication from an encrypted DB cluster.

The pre-signed URL must be a valid request for the `CreateDBCluster` API action that can be executed in the source AWS Region that contains the encrypted DB cluster to be copied.

The pre-signed URL request must contain the following parameter values:
- **KmsKeyId** - The AWS KMS key identifier for the key to use to encrypt the copy of the DB cluster in the destination AWS Region. This should refer to the same KMS key for both the `CreateDBCluster` action that is called in the destination AWS Region, and the action contained in the pre-signed URL.
- **DestinationRegion** - The name of the AWS Region that Read Replica will be created in.
- **ReplicationSourceIdentifier** - The DB cluster identifier for the encrypted DB cluster to be copied. This identifier must be in the Amazon Resource Name (ARN) format for the source AWS Region. For example, if you are copying an encrypted DB cluster from the us-west-2 AWS Region, then your `ReplicationSourceIdentifier` would look like Example: `arn:aws:rds:us-west-2:123456789012:cluster:neptune-cluster1`.
To learn how to generate a Signature Version 4 signed request, see Authenticating Requests: Using Query Parameters (AWS Signature Version 4) and Signature Version 4 Signing Process.

**Type:** String  
**Required:** No

**ReplicationSourceIdentifier**

The Amazon Resource Name (ARN) of the source DB instance or DB cluster if this DB cluster is created as a Read Replica.

**Type:** String  
**Required:** No

**StorageEncrypted**

Specifies whether the DB cluster is encrypted.

**Type:** Boolean  
**Required:** No

**Tags.Tag.N**

A list of tags. For more information, see Tagging Amazon Neptune Resources.

**Type:** Array of Tag (p. 358) objects  
**Required:** No

**VpcSecurityGroupIds.VpcSecurityGroupId.N**

A list of EC2 VPC security groups to associate with this DB cluster.

**Type:** Array of strings  
**Required:** No

### Response Elements

The following element is returned by the service.

**DBCluster**

Contains the details of an Amazon Neptune DB cluster.

This data type is used as a response element in the DescribeDBClusters (p. 210) action.

**Type:** DBCluster (p. 304) object

### Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

**DBClusterAlreadyExistsFault**

User already has a DB cluster with the given identifier.

HTTP Status Code: 400
**DBClusterNotFoundFault**

*DBClusterIdentifier* does not refer to an existing DB cluster.

HTTP Status Code: 404

**DBClusterParameterGroupNotFound**

*DBClusterParameterGroupName* does not refer to an existing DB Cluster parameter group.

HTTP Status Code: 404

**DBClusterQuotaExceededFault**

User attempted to create a new DB cluster and the user has already reached the maximum allowed DB cluster quota.

HTTP Status Code: 403

**DBInstanceNotFound**

*DBInstanceIdentifier* does not refer to an existing DB instance.

HTTP Status Code: 404

**DBSubnetGroupDoesNotCoverEnoughAZs**

Subnets in the DB subnet group should cover at least two Availability Zones unless there is only one Availability Zone.

HTTP Status Code: 400

**DBSubnetGroupNotFoundFault**

*DBSubnetGroupName* does not refer to an existing DB subnet group.

HTTP Status Code: 404

**InsufficientStorageClusterCapacity**

There is insufficient storage available for the current action. You may be able to resolve this error by updating your subnet group to use different Availability Zones that have more storage available.

HTTP Status Code: 400

**InvalidDBClusterStateFault**

The DB cluster is not in a valid state.

HTTP Status Code: 400

**InvalidDBInstanceState**

The specified DB instance is not in the *available* state.

HTTP Status Code: 400

**InvalidDBSubnetGroupStateFault**

The DB subnet group cannot be deleted because it is in use.

HTTP Status Code: 400

**InvalidSubnet**

The requested subnet is invalid, or multiple subnets were requested that are not all in a common VPC.

HTTP Status Code: 400
InvalidVPCNetworkStateFault

DB subnet group does not cover all Availability Zones after it is created because users' change.

HTTP Status Code: 400

KMSKeyNotAccessibleFault

Error accessing KMS key.

HTTP Status Code: 400

StorageQuotaExceeded

Request would result in user exceeding the allowed amount of storage available across all DB instances.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
CreateDBClusterParameterGroup

Creates a new DB cluster parameter group.

Parameters in a DB cluster parameter group apply to all of the instances in a DB cluster.

A DB cluster parameter group is initially created with the default parameters for the database engine used by instances in the DB cluster. To provide custom values for any of the parameters, you must modify the group after creating it using ModifyDBClusterParameterGroup (p. 255). Once you've created a DB cluster parameter group, you need to associate it with your DB cluster using ModifyDBCluster (p. 250). When you associate a new DB cluster parameter group with a running DB cluster, you need to reboot the DB instances in the DB cluster without failover for the new DB cluster parameter group and associated settings to take effect.

**Important**
After you create a DB cluster parameter group, you should wait at least 5 minutes before creating your first DB cluster that uses that DB cluster parameter group as the default parameter group. This allows Amazon Neptune to fully complete the create action before the DB cluster parameter group is used as the default for a new DB cluster. This is especially important for parameters that are critical when creating the default database for a DB cluster, such as the character set for the default database defined by the `character_set_database` parameter. You can use the `Parameter Groups` option of the Amazon Neptune console or the `DescribeDBClusterParameters` (p. 208) command to verify that your DB cluster parameter group has been created or modified.

**Request Parameters**

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

**DBClusterParameterGroupName**

The name of the DB cluster parameter group.

Constraints:

- Must match the name of an existing DBClusterParameterGroup.

  **Note**
  This value is stored as a lowercase string.

  Type: String

  Required: Yes

**DBParameterGroupFamily**

The DB cluster parameter group family name. A DB cluster parameter group can be associated with one and only one DB cluster parameter group family, and can be applied only to a DB cluster running a database engine and engine version compatible with that DB cluster parameter group family.

Type: String

Required: Yes

**Description**

The description for the DB cluster parameter group.

Type: String
**Tags.Tag.N**

A list of tags. For more information, see [Tagging Amazon Neptune Resources](#).

Type: Array of [Tag](#) objects

Required: No

**Response Elements**

The following element is returned by the service.

**DBClusterParameterGroup**

Contains the details of an Amazon Neptune DB cluster parameter group.

This data type is used as a response element in the [DescribeDBClusterParameterGroups](#) action.

Type: [DBClusterParameterGroup](#) object

**Errors**

For information about the errors that are common to all actions, see [Common Errors](#).

**DBParameterGroupAlreadyExists**

A DB parameter group with the same name exists.

HTTP Status Code: 400

**DBParameterGroupQuotaExceeded**

Request would result in user exceeding the allowed number of DB parameter groups.

HTTP Status Code: 400

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
CreateDBClusterSnapshot

Creates a snapshot of a DB cluster.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

DBClusterIdentifier

The identifier of the DB cluster to create a snapshot for. This parameter is not case-sensitive.

Constraints:
- Must match the identifier of an existing DBCluster.

Example: my-cluster1

Type: String

Required: Yes

DBClusterSnapshotIdentifier

The identifier of the DB cluster snapshot. This parameter is stored as a lowercase string.

Constraints:
- Must contain from 1 to 63 letters, numbers, or hyphens.
- First character must be a letter.
- Cannot end with a hyphen or contain two consecutive hyphens.

Example: my-cluster1-snapshot1

Type: String

Required: Yes

Tags.Tag.N

The tags to be assigned to the DB cluster snapshot.

Type: Array of Tag (p. 358) objects

Required: No

Response Elements

The following element is returned by the service.

DBClusterSnapshot

Contains the details for an Amazon Neptune DB cluster snapshot.

This data type is used as a response element in the DescribeDBClusterSnapshots (p. 214) action.

Type: DBClusterSnapshot (p. 313) object
Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

**DBClusterNotFoundFault**

*DBClusterIdentifier* does not refer to an existing DB cluster.

HTTP Status Code: 404

**DBClusterSnapshotAlreadyExistsFault**

User already has a DB cluster snapshot with the given identifier.

HTTP Status Code: 400

**InvalidDBClusterSnapshotStateFault**

The supplied value is not a valid DB cluster snapshot state.

HTTP Status Code: 400

**InvalidDBClusterStateFault**

The DB cluster is not in a valid state.

HTTP Status Code: 400

**SnapshotQuotaExceeded**

Request would result in user exceeding the allowed number of DB snapshots.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
CreateDBInstance

Creates a new DB instance.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

AllocatedStorage

The amount of storage (in gibibytes) to allocate for the DB instance.

Type: Integer

Not applicable. Neptune cluster volumes automatically grow as the amount of data in your database increases, though you are only charged for the space that you use in a Neptune cluster volume.

Type: Integer

Required: No

AutoMinorVersionUpgrade

Indicates that minor engine upgrades are applied automatically to the DB instance during the maintenance window.

Default: true

Type: Boolean

Required: No

AvailabilityZone

The EC2 Availability Zone that the DB instance is created in.

Default: A random, system-chosen Availability Zone in the endpoint’s AWS Region.

Example: us-east-1d

Constraint: The AvailabilityZone parameter can’t be specified if the MultiAZ parameter is set to true. The specified Availability Zone must be in the same AWS Region as the current endpoint.

Type: String

Required: No

BackupRetentionPeriod

The number of days for which automated backups are retained.

Not applicable. The retention period for automated backups is managed by the DB cluster. For more information, see CreateDBCluster (p. 164).

Default: 1

Constraints:

- Must be a value from 0 to 35
- Cannot be set to 0 if the DB instance is a source to Read Replicas

Type: Integer
**CreateDBInstance**

- **Required:** No

- **CharacterSetName**
  - Indicates that the DB instance should be associated with the specified CharacterSet.
  - Not applicable. The character set is managed by the DB cluster. For more information, see `CreateDBCluster (p. 164)`.
  - **Type:** String
  - **Required:** No

- **CopyTagsToSnapshot**
  - True to copy all tags from the DB instance to snapshots of the DB instance, and otherwise false. The default is false.
  - **Type:** Boolean
  - **Required:** No

- **DBClusterIdentifier**
  - The identifier of the DB cluster that the instance will belong to.
  - For information on creating a DB cluster, see `CreateDBCluster (p. 164)`.
  - **Type:** String
  - **Type:** String
  - **Required:** No

- **DBInstanceClass**
  - The compute and memory capacity of the DB instance, for example, `db.m4.large`. Not all DB instance classes are available in all AWS Regions.
  - **Type:** String
  - **Required:** Yes

- **DBInstanceIdentifier**
  - The DB instance identifier. This parameter is stored as a lowercase string.
  - **Constraints:**
    - Must contain from 1 to 63 letters, numbers, or hyphens.
    - First character must be a letter.
    - Cannot end with a hyphen or contain two consecutive hyphens.
  - **Example:** `mydbinstance`
  - **Type:** String
  - **Required:** Yes

- **DBName**
  - The database name.
  - **Type:** String
  - **Type:** String
DBParameterGroupName

The name of the DB parameter group to associate with this DB instance. If this argument is omitted, the default DBParameterGroup for the specified engine is used.

Constraints:
- Must be 1 to 255 letters, numbers, or hyphens.
- First character must be a letter
- Cannot end with a hyphen or contain two consecutive hyphens

Type: String

DBSecurityGroups.DBSecurityGroupName.N

A list of DB security groups to associate with this DB instance.

Default: The default DB security group for the database engine.

Type: Array of strings

DBSubnetGroupName

A DB subnet group to associate with this DB instance.

If there is no DB subnet group, then it is a non-VPC DB instance.

Type: String

Domain

Specify the Active Directory Domain to create the instance in.

Type: String

DomainIAMRoleName

Specify the name of the IAM role to be used when making API calls to the Directory Service.

Type: String

EnableCloudwatchLogsExports.member.N

The list of log types that need to be enabled for exporting to CloudWatch Logs.

Type: Array of strings

EnableIAMDatabaseAuthentication

True to enable AWS Identity and Access Management (IAM) authentication for Neptune.

Default: false
EnablePerformanceInsights

Type: Boolean
Required: No

True to enable Performance Insights for the DB instance, and otherwise false.

Engine

Type: String
Required: Yes

The name of the database engine to be used for this instance.

Valid Values: neptune

EngineVersion

Type: String
Required: No

The version number of the database engine to use.

Iops

Type: Integer
Required: No

The amount of Provisioned IOPS (input/output operations per second) to be initially allocated for the DB instance.

KmsKeyId

Type: String
Required: No

The AWS KMS key identifier for an encrypted DB instance.

The KMS key identifier is the Amazon Resource Name (ARN) for the KMS encryption key. If you are creating a DB instance with the same AWS account that owns the KMS encryption key used to encrypt the new DB instance, then you can use the KMS key alias instead of the ARN for the KMS encryption key.

Not applicable. The KMS key identifier is managed by the DB cluster. For more information, see CreateDBCluster (p. 164).

If the StorageEncrypted parameter is true, and you do not specify a value for the KmsKeyId parameter, then Amazon Neptune will use your default encryption key. AWS KMS creates the default encryption key for your AWS account. Your AWS account has a different default encryption key for each AWS Region.

LicenseModel

Type: String
Required: No

License model information for this DB instance.

Valid values: license-included | bring-your-own-license | general-public-license
CreateDBInstance

Type: String
Required: No

MasterUsername
The name for the master user. Not used.
Type: String
Required: No

MasterUserPassword
The password for the master user. The password can include any printable ASCII character except "/", "", or ".
Not used.
Type: String
Required: No

MonitoringInterval
The interval, in seconds, between points when Enhanced Monitoring metrics are collected for the DB instance. To disable collecting Enhanced Monitoring metrics, specify 0. The default is 0.

If MonitoringRoleArn is specified, then you must also set MonitoringInterval to a value other than 0.

Valid Values: 0, 1, 5, 10, 15, 30, 60
Type: Integer
Required: No

MonitoringRoleArn
The ARN for the IAM role that permits Neptune to send enhanced monitoring metrics to Amazon CloudWatch Logs. For example, arn:aws:iam:123456789012:role/emaccess.

If MonitoringInterval is set to a value other than 0, then you must supply a MonitoringRoleArn value.

Type: String
Required: No

MultiAZ
Specifies if the DB instance is a Multi-AZ deployment. You can't set the AvailabilityZone parameter if the MultiAZ parameter is set to true.

Type: Boolean
Required: No

OptionGroupName
Indicates that the DB instance should be associated with the specified option group.

Permanent options, such as the TDE option for Oracle Advanced Security TDE, can't be removed from an option group, and that option group can't be removed from a DB instance once it is associated with a DB instance.
Type: String
Required: No

**PerformanceInsightsKMSKeyId**

The AWS KMS key identifier for encryption of Performance Insights data. The KMS key ID is the Amazon Resource Name (ARN), KMS key identifier, or the KMS key alias for the KMS encryption key.

Type: String
Required: No

**Port**

The port number on which the database accepts connections.

Not applicable. The port is managed by the DB cluster. For more information, see `CreateDBCluster (p. 164)`.

Default: 8182

Type: Integer

Type: Integer

Required: No

**PreferredBackupWindow**

The daily time range during which automated backups are created.

Not applicable. The daily time range for creating automated backups is managed by the DB cluster. For more information, see `CreateDBCluster (p. 164)`.

Type: String
Required: No

**PreferredMaintenanceWindow**

The time range each week during which system maintenance can occur, in Universal Coordinated Time (UTC).


The default is a 30-minute window selected at random from an 8-hour block of time for each AWS Region, occurring on a random day of the week.

Valid Days: Mon, Tue, Wed, Thu, Fri, Sat, Sun.

Constraints: Minimum 30-minute window.

Type: String
Required: No

**PromotionTier**

A value that specifies the order in which an Read Replica is promoted to the primary instance after a failure of the existing primary instance.

Default: 1

Valid Values: 0 - 15
Type: Integer
Required: No

**PubliclyAccessible**

*This parameter has been deprecated.*

This parameter is not supported.

Type: Boolean
Required: No

**StorageEncrypted**

Specifies whether the DB instance is encrypted.

Not applicable. The encryption for DB instances is managed by the DB cluster. For more information, see CreateDBCluster (p. 164).

Default: false
Type: Boolean
Required: No

**StorageType**

Specifies the storage type to be associated with the DB instance.

Not applicable. Storage is managed by the DB Cluster.

Type: String
Required: No

**Tags.Tag.N**

A list of tags. For more information, see Tagging Amazon Neptune Resources.

Type: Array of Tag (p. 358) objects
Required: No

**TdeCredentialArn**

The ARN from the key store with which to associate the instance for TDE encryption.

Type: String
Required: No

**TdeCredentialPassword**

The password for the given ARN from the key store in order to access the device.

Type: String
Required: No

**Timezone**

The time zone of the DB instance.

Type: String
VpcSecurityGroupIds.VpcSecurityGroupId.N

A list of EC2 VPC security groups to associate with this DB instance.

Not applicable. The associated list of EC2 VPC security groups is managed by the DB cluster. For more information, see CreateDBCluster (p. 164).

Default: The default EC2 VPC security group for the DB subnet group's VPC.

Type: Array of strings

Required: No

Response Elements

The following element is returned by the service.

DBInstance

Contains the details of an Amazon Neptune DB instance.

This data type is used as a response element in the DescribeDBInstances (p. 220) action.

Type: DBInstance (p. 320) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

AuthorizationNotFound

Specified CIDRIP or EC2 security group is not authorized for the specified DB security group.

Neptune may not also be authorized via IAM to perform necessary actions on your behalf.

HTTP Status Code: 404

DBClusterNotFoundFault

DBClusterIdentifier does not refer to an existing DB cluster.

HTTP Status Code: 404

DBInstanceAlreadyExists

User already has a DB instance with the given identifier.

HTTP Status Code: 400

DBParameterGroupNotFound

DBParameterGroupName does not refer to an existing DB parameter group.

HTTP Status Code: 404

DBSecurityGroupNotFound

DBSecurityGroupName does not refer to an existing DB security group.

HTTP Status Code: 404
DBSubnetGroupDoesNotExist
Subnets in the DB subnet group should cover at least two Availability Zones unless there is only one Availability Zone.
HTTP Status Code: 400

DBSubnetGroupNameNotFoundFault
DBSubnetGroupName does not refer to an existing DB subnet group.
HTTP Status Code: 404

DomainNameNotFoundFault
Domain does not refer to an existing Active Directory Domain.
HTTP Status Code: 404

InstanceQuotaExceeded
Request would result in user exceeding the allowed number of DB instances.
HTTP Status Code: 400

InsufficientDBInstanceCapacity
Specified DB instance class is not available in the specified Availability Zone.
HTTP Status Code: 400

InvalidDBClusterStateFault
The DB cluster is not in a valid state.
HTTP Status Code: 400

InvalidSubnet
The requested subnet is invalid, or multiple subnets were requested that are not all in a common VPC.
HTTP Status Code: 400

InvalidVPCNetworkStateFault
DB subnet group does not cover all Availability Zones after it is created because users' change.
HTTP Status Code: 400

KMSKeyNotAccessibleFault
Error accessing KMS key.
HTTP Status Code: 400

OptionGroupNameNotFoundFault
HTTP Status Code: 404

ProvisionedIopsNotAvailableInAZFault
Provisioned IOPS not available in the specified Availability Zone.
HTTP Status Code: 400

StorageQuotaExceeded
Request would result in user exceeding the allowed amount of storage available across all DB instances.
HTTP Status Code: 400

**StorageTypeNotSupported**

*StorageType* specified cannot be associated with the DB Instance.

HTTP Status Code: 400

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
CreateDBParameterGroup

Creates a new DB parameter group.

A DB parameter group is initially created with the default parameters for the database engine used by the DB instance. To provide custom values for any of the parameters, you must modify the group after creating it using ModifyDBParameterGroup. Once you've created a DB parameter group, you need to associate it with your DB instance using ModifyDBInstance. When you associate a new DB parameter group with a running DB instance, you need to reboot the DB instance without failover for the new DB parameter group and associated settings to take effect.

Important
After you create a DB parameter group, you should wait at least 5 minutes before creating your first DB instance that uses that DB parameter group as the default parameter group. This allows Amazon Neptune to fully complete the create action before the parameter group is used as the default for a new DB instance. This is especially important for parameters that are critical when creating the default database for a DB instance, such as the character set for the default database defined by the character_set_database parameter. You can use the Parameter Groups option of the Amazon Neptune console or the DescribeDBParameters command to verify that your DB parameter group has been created or modified.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

DBParameterGroupFamily

The DB parameter group family name. A DB parameter group can be associated with one and only one DB parameter group family, and can be applied only to a DB instance running a database engine and engine version compatible with that DB parameter group family.

Type: String
Required: Yes

DBParameterGroupName

The name of the DB parameter group.

Constraints:
• Must be 1 to 255 letters, numbers, or hyphens.
• First character must be a letter
• Cannot end with a hyphen or contain two consecutive hyphens

Note
This value is stored as a lowercase string.

Type: String
Required: Yes

Description

The description for the DB parameter group.

Type: String
Required: Yes
Tags.Tag.N

A list of tags. For more information, see Tagging Amazon Neptune Resources.

Type: Array of Tag (p. 358) objects

Required: No

Response Elements

The following element is returned by the service.

DBParameterGroup

Contains the details of an Amazon Neptune DB parameter group.

This data type is used as a response element in the DescribeDBParameterGroups (p. 222) action.

Type: DBParameterGroup (p. 328) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

DBParameterGroupAlreadyExists

A DB parameter group with the same name exists.

HTTP Status Code: 400

DBParameterGroupQuotaExceeded

Request would result in user exceeding the allowed number of DB parameter groups.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
CreateDBSubnetGroup

Creates a new DB subnet group. DB subnet groups must contain at least one subnet in at least two AZs in the AWS Region.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

DBSubnetGroupDescription

The description for the DB subnet group.

Type: String

Required: Yes

DBSubnetGroupName

The name for the DB subnet group. This value is stored as a lowercase string.

Constraints: Must contain no more than 255 letters, numbers, periods, underscores, spaces, or hyphens. Must not be default.

Example: mySubnetgroup

Type: String

Required: Yes

SubnetIds.SubnetIdentifier.N

The EC2 Subnet IDs for the DB subnet group.

Type: Array of strings

Required: Yes

Tags.Tag.N

A list of tags. For more information, see Tagging Amazon Neptune Resources.

Type: Array of Tag (p. 358) objects

Required: No

Response Elements

The following element is returned by the service.

DBSubnetGroup

Contains the details of an Amazon Neptune DB subnet group.

This data type is used as a response element in the DescribeDBSubnetGroups (p. 226) action.

Type: DBSubnetGroup (p. 331) object
Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

**DBSubnetGroupAlreadyExists**

*DBSubnetGroupName* is already used by an existing DB subnet group.

HTTP Status Code: 400

**DBSubnetGroupDoesNotCoverEnoughAZs**

Subnets in the DB subnet group should cover at least two Availability Zones unless there is only one Availability Zone.

HTTP Status Code: 400

**DBSubnetGroupQuotaExceeded**

Request would result in user exceeding the allowed number of DB subnet groups.

HTTP Status Code: 400

**DBSubnetQuotaExceededFault**

Request would result in user exceeding the allowed number of subnets in a DB subnet groups.

HTTP Status Code: 400

**InvalidSubnet**

The requested subnet is invalid, or multiple subnets were requested that are not all in a common VPC.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
CreateEventSubscription

Creates an event notification subscription. This action requires a topic ARN (Amazon Resource Name) created by either the Neptune console, the SNS console, or the SNS API. To obtain an ARN with SNS, you must create a topic in Amazon SNS and subscribe to the topic. The ARN is displayed in the SNS console.

You can specify the type of source (SourceType) you want to be notified of, provide a list of Neptune sources (SourceIds) that triggers the events, and provide a list of event categories (EventCategories) for events you want to be notified of. For example, you can specify SourceType = db-instance, SourceIds = mydbinstance1, mydbinstance2 and EventCategories = Availability, Backup.

If you specify both the SourceType and SourceIds, such as SourceType = db-instance and SourceIdentifier = myDBInstance1, you are notified of all the db-instance events for the specified source. If you specify a SourceType but do not specify a SourceIdentifier, you receive notice of the events for that source type for all your Neptune sources. If you do not specify either the SourceType nor the SourceIdentifier, you are notified of events generated from all Neptune sources belonging to your customer account.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

Enabled

A Boolean value; set to true to activate the subscription, set to false to create the subscription but not activate it.

Type: Boolean

Required: No

EventCategories.EventCategory.N

A list of event categories for a SourceType that you want to subscribe to. You can see a list of the categories for a given SourceType by using the DescribeEventCategories action.

Type: Array of strings

Required: No

SnsTopicArn

The Amazon Resource Name (ARN) of the SNS topic created for event notification. The ARN is created by Amazon SNS when you create a topic and subscribe to it.

Type: String

Required: Yes

SourceIds.SourceId.N

The list of identifiers of the event sources for which events are returned. If not specified, then all sources are included in the response. An identifier must begin with a letter and must contain only ASCII letters, digits, and hyphens; it can't end with a hyphen or contain two consecutive hyphens.

Constraints:
- If SourceIds are supplied, SourceType must also be provided.
- If the source type is a DB instance, then a DBInstanceIdentifier must be supplied.
- If the source type is a DB security group, a DBSecurityGroupGroupName must be supplied.
- If the source type is a DB parameter group, a DBParameterGroupName must be supplied.
CreateEventSubscription

• If the source type is a DB snapshot, a DBSnapshotIdentifier must be supplied.

Type: Array of strings
Required: No

SourceType

The type of source that is generating the events. For example, if you want to be notified of events generated by a DB instance, you would set this parameter to db-instance. If this value is not specified, all events are returned.

Valid values: db-instance | db-cluster | db-parameter-group | db-security-group | db-snapshot | db-cluster-snapshot

Type: String
Required: No

SubscriptionName

The name of the subscription.
Constraints: The name must be less than 255 characters.

Type: String
Required: Yes

Tags.Tag.N

A list of tags. For more information, see Tagging Amazon Neptune Resources.

Type: Array of Tag (p. 358) objects
Required: No

Response Elements

The following element is returned by the service.

EventSubscription

Contains the results of a successful invocation of the DescribeEventSubscriptions (p. 237) action.

Type: EventSubscription (p. 340) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

EventSubscriptionQuotaExceeded

HTTP Status Code: 400

SNSInvalidTopic

HTTP Status Code: 400

SNSNoAuthorization

HTTP Status Code: 400
SNSTopicArnNotFound

HTTP Status Code: 404

SourceNotFound

HTTP Status Code: 404

SubscriptionAlreadyExist

HTTP Status Code: 400

SubscriptionCategoryNotFound

HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DeleteDBCluster

The DeleteDBCluster action deletes a previously provisioned DB cluster. When you delete a DB cluster, all automated backups for that DB cluster are deleted and can't be recovered. Manual DB cluster snapshots of the specified DB cluster are not deleted.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

DBClusterIdentifier

The DB cluster identifier for the DB cluster to be deleted. This parameter isn't case-sensitive.

- Constraints:
  - Must match an existing DBClusterIdentifier.

- Type: String

- Required: Yes

FinalDBSnapshotIdentifier

The DB cluster snapshot identifier of the new DB cluster snapshot created when SkipFinalSnapshot is set to false.

- Note
  - Specifying this parameter and also setting the SkipFinalSnapshot parameter to true results in an error.

- Constraints:
  - Must be 1 to 255 letters, numbers, or hyphens.
  - First character must be a letter
  - Cannot end with a hyphen or contain two consecutive hyphens

- Type: String

- Required: No

SkipFinalSnapshot

Determines whether a final DB cluster snapshot is created before the DB cluster is deleted. If true is specified, no DB cluster snapshot is created. If false is specified, a DB cluster snapshot is created before the DB cluster is deleted.

- Note
  - You must specify a FinalDBSnapshotIdentifier parameter if SkipFinalSnapshot is false.

- Default: false

- Type: Boolean

- Required: No

Response Elements

The following element is returned by the service.
**DBCluster**

Contains the details of an Amazon Neptune DB cluster.

This data type is used as a response element in the **DescribeDBClusters (p. 210)** action.

Type: **DBCluster (p. 304)** object

**Errors**

For information about the errors that are common to all actions, see **Common Errors (p. 363)**.

**DBClusterNotFoundFault**

*DBClusterIdentifier* does not refer to an existing DB cluster.

HTTP Status Code: 404

**DBClusterSnapshotAlreadyExistsFault**

User already has a DB cluster snapshot with the given identifier.

HTTP Status Code: 400

**InvalidDBClusterSnapshotStateFault**

The supplied value is not a valid DB cluster snapshot state.

HTTP Status Code: 400

**InvalidDBClusterStateFault**

The DB cluster is not in a valid state.

HTTP Status Code: 400

**SnapshotQuotaExceeded**

Request would result in user exceeding the allowed number of DB snapshots.

HTTP Status Code: 400

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DeleteDBClusterParameterGroup

Deletes a specified DB cluster parameter group. The DB cluster parameter group to be deleted can't be associated with any DB clusters.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

DBClusterParameterGroupName

The name of the DB cluster parameter group.

Constraints:
- Must be the name of an existing DB cluster parameter group.
- You can't delete a default DB cluster parameter group.
- Cannot be associated with any DB clusters.

Type: String

Required: Yes

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

DBParameterGroupNotFound

DBParameterGroupName does not refer to an existing DB parameter group.

HTTP Status Code: 404

InvalidDBParameterGroupState

The DB parameter group is in use or is in an invalid state. If you are attempting to delete the parameter group, you cannot delete it when the parameter group is in this state.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DeleteDBClusterSnapshot

Deletes a DB cluster snapshot. If the snapshot is being copied, the copy operation is terminated.

**Note**
The DB cluster snapshot must be in the `available` state to be deleted.

**Request Parameters**

For information about the parameters that are common to all actions, see [Common Parameters](#).

**DBClusterSnapshotIdentifier**

- The identifier of the DB cluster snapshot to delete.
- Constraints: Must be the name of an existing DB cluster snapshot in the `available` state.
- Type: String
- Required: Yes

**Response Elements**

The following element is returned by the service.

**DBClusterSnapshot**

- Contains the details for an Amazon Neptune DB cluster snapshot
- This data type is used as a response element in the [DescribeDBClusterSnapshots](#) action.
- Type: [DBClusterSnapshot](#) object

**Errors**

For information about the errors that are common to all actions, see [Common Errors](#).

**DBClusterSnapshotNotFoundFault**

- `DBClusterSnapshotIdentifier` does not refer to an existing DB cluster snapshot.
- HTTP Status Code: 404

**InvalidDBClusterSnapshotStateFault**

- The supplied value is not a valid DB cluster snapshot state.
- HTTP Status Code: 400

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET

---

API Version 2017-11-29

196
DeleteDBClusterSnapshot

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DeleteDBInstance

The DeleteDBInstance action deletes a previously provisioned DB instance. When you delete a DB instance, all automated backups for that instance are deleted and can't be recovered. Manual DB snapshots of the DB instance to be deleted by DeleteDBInstance are not deleted.

If you request a final DB snapshot the status of the Amazon Neptune DB instance is deleting until the DB snapshot is created. The API action DescribeDBInstance is used to monitor the status of this operation. The action can't be canceled or reverted once submitted.

Note that when a DB instance is in a failure state and has a status of failed, incompatible-restore, or incompatible-network, you can only delete it when the SkipFinalSnapshot parameter is set to true.

If the specified DB instance is part of a DB cluster, you can't delete the DB instance if both of the following conditions are true:

- The DB cluster is a Read Replica of another DB cluster.
- The DB instance is the only instance in the DB cluster.

To delete a DB instance in this case, first call the PromoteReadReplicaDBCluster (p. 276) API action to promote the DB cluster so it's no longer a Read Replica. After the promotion completes, then call the DeleteDBInstance API action to delete the final instance in the DB cluster.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

DBInstanceIdentifier

The DB instance identifier for the DB instance to be deleted. This parameter isn't case-sensitive.

Constraints:
- Must match the name of an existing DB instance.

Type: String

Required: Yes

FinalDBSnapshotIdentifier

The DBSnapshotIdentifier of the new DBSnapshot created when SkipFinalSnapshot is set to false.

Note

Specifying this parameter and also setting the SkipFinalSnapshot parameter to true results in an error.

Constraints:
- Must be 1 to 255 letters or numbers.
- First character must be a letter
- Cannot end with a hyphen or contain two consecutive hyphens
- Cannot be specified when deleting a Read Replica.

Type: String

Required: No
SkipFinalSnapshot

Determines whether a final DB snapshot is created before the DB instance is deleted. If true is specified, no DBSnapshot is created. If false is specified, a DB snapshot is created before the DB instance is deleted.

Note that when a DB instance is in a failure state and has a status of 'failed', 'incompatible-restore', or 'incompatible-network', it can only be deleted when the SkipFinalSnapshot parameter is set to "true".

Specify true when deleting a Read Replica.

Note

The FinalDBSnapshotIdentifier parameter must be specified if SkipFinalSnapshot is false.

Default: false

Type: Boolean

Required: No

Response Elements

The following element is returned by the service.

DBInstance

Contains the details of an Amazon Neptune DB instance.

This data type is used as a response element in the DescribeDBInstances (p. 220) action.

Type: DBInstance (p. 320) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

DBInstanceNotFound

DBInstanceIdentifier does not refer to an existing DB instance.

HTTP Status Code: 404

DBSnapshotAlreadyExists

DBSnapshotIdentifier is already used by an existing snapshot.

HTTP Status Code: 400

InvalidDBClusterStateFault

The DB cluster is not in a valid state.

HTTP Status Code: 400

InvalidDBInstanceState

The specified DB instance is not in the available state.

HTTP Status Code: 400
SnapshotQuotaExceeded

Request would result in user exceeding the allowed number of DB snapshots.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DeleteDBParameterGroup

Deletes a specified DBParameterGroup. The DBParameterGroup to be deleted can't be associated with any DB instances.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

DBParameterGroupName

The name of the DB parameter group.

Constraints:

- Must be the name of an existing DB parameter group
- You can't delete a default DB parameter group
- Cannot be associated with any DB instances

Type: String

Required: Yes

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

DBParameterGroupNotFound

DBParameterGroupName does not refer to an existing DB parameter group.

HTTP Status Code: 404

InvalidDBParameterGroupState

The DB parameter group is in use or is in an invalid state. If you are attempting to delete the parameter group, you cannot delete it when the parameter group is in this state.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DeleteDBSubnetGroup

Deletes a DB subnet group.

**Note**
The specified database subnet group must not be associated with any DB instances.

**Request Parameters**

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

**DBSubnetGroupName**

The name of the database subnet group to delete.

**Note**
You can't delete the default subnet group.

Constraints:

Constraints: Must match the name of an existing DBSubnetGroup. Must not be default.

Example: mySubnetgroup

Type: String

Required: Yes

**Errors**

For information about the errors that are common to all actions, see Common Errors (p. 363).

**DBSubnetGroupNotFoundFault**

**DBSubnetGroupName** does not refer to an existing DB subnet group.

HTTP Status Code: 404

**InvalidDBSubnetGroupStateFault**

The DB subnet group cannot be deleted because it is in use.

HTTP Status Code: 400

**InvalidDBSubnetStateFault**

The DB subnet is not in the *available* state.

HTTP Status Code: 400

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DeleteEventSubscription

Deletes an event notification subscription.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

SubscriptionName

The name of the event notification subscription you want to delete.

Type: String
Required: Yes

Response Elements

The following element is returned by the service.

EventSubscription

Contains the results of a successful invocation of the DescribeEventSubscriptions (p. 237) action.

Type: EventSubscription (p. 340) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

InvalidEventSubscriptionState

HTTP Status Code: 400

SubscriptionNotFound

HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeDBClusterParameterGroups

Returns a list of DBClusterParameterGroup descriptions. If a DBClusterParameterGroupName parameter is specified, the list will contain only the description of the specified DB cluster parameter group.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

DBClusterParameterGroupName

The name of a specific DB cluster parameter group to return details for.

Constraints:

- If supplied, must match the name of an existing DBClusterParameterGroup.

Type: String

Required: No

Filters.Filter.N

This parameter is not currently supported.

Type: Array of Filter (p. 342) objects

Required: No

Marker

An optional pagination token provided by a previous DescribeDBClusterParameterGroups request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by MaxRecords.

Type: String

Required: No

MaxRecords

The maximum number of records to include in the response. If more records exist than the specified MaxRecords value, a pagination token called a marker is included in the response so that the remaining results can be retrieved.

Default: 100

Constraints: Minimum 20, maximum 100.

Type: Integer

Required: No

Response Elements

The following elements are returned by the service.

DBClusterParameterGroups.DBClusterParameterGroup.N

A list of DB cluster parameter groups.
Type: Array of DBClusterParameterGroup (p. 311) objects

Marker

An optional pagination token provided by a previous DescribeDBClusterParameterGroups request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by MaxRecords.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

DBParameterGroupNotFound

DBParameterGroupName does not refer to an existing DB parameter group.

HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeDBClusterParameters

Returns the detailed parameter list for a particular DB cluster parameter group.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

**DBClusterParameterGroupName**

The name of a specific DB cluster parameter group to return parameter details for.

Constraints:

- If supplied, must match the name of an existing DBClusterParameterGroup.

  Type: String

  Required: Yes

**Filters.Filter.N**

This parameter is not currently supported.

  Type: Array of Filter (p. 342) objects

  Required: No

**Marker**

An optional pagination token provided by a previous DescribeDBClusterParameters request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by MaxRecords.

  Type: String

  Required: No

**MaxRecords**

The maximum number of records to include in the response. If more records exist than the specified MaxRecords value, a pagination token called a marker is included in the response so that the remaining results can be retrieved.

  Default: 100

  Constraints: Minimum 20, maximum 100.

  Type: Integer

  Required: No

**Source**

A value that indicates to return only parameters for a specific source. Parameter sources can be engine, service, or customer.

  Type: String

  Required: No
Response Elements

The following elements are returned by the service.

Marker

An optional pagination token provided by a previous DescribeDBClusterParameters request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by MaxRecords.

Type: String

Parameters.Parameter.N

Provides a list of parameters for the DB cluster parameter group.

Type: Array of Parameter (p. 347) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

DBParameterGroupNotFound

DBParameterGroupName does not refer to an existing DB parameter group.

HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeDBClusters

Returns information about provisioned DB clusters. This API supports pagination.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

**DBClusterIdentifier**

The user-supplied DB cluster identifier. If this parameter is specified, information from only the specific DB cluster is returned. This parameter isn't case-sensitive.

- **Constraints:**
  - If supplied, must match an existing DBClusterIdentifier.

  - **Type:** String
  - **Required:** No

**Filters.Filter.N**

A filter that specifies one or more DB clusters to describe.

- **Supported filters:**
  - `db-cluster-id` - Accepts DB cluster identifiers and DB cluster Amazon Resource Names (ARNs). The results list will only include information about the DB clusters identified by these ARNs.

  - **Type:** Array of Filter (p. 342) objects
  - **Required:** No

**Marker**

An optional pagination token provided by a previous DescribeDBClusters (p. 210) request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by MaxRecords.

- **Type:** String
- **Required:** No

**MaxRecords**

The maximum number of records to include in the response. If more records exist than the specified MaxRecords value, a pagination token called a marker is included in the response so that the remaining results can be retrieved.

- **Default:** 100
- **Constraints:** Minimum 20, maximum 100.

- **Type:** Integer
- **Required:** No

Response Elements

The following elements are returned by the service.
DescribeDBClusters

DBClusters.DBCluster.N
Contains a list of DB clusters for the user.
Type: Array of DBCluster (p. 304) objects

Marker
A pagination token that can be used in a subsequent DescribeDBClusters request.
Type: String

Errors
For information about the errors that are common to all actions, see Common Errors (p. 363).

DBClusterNotFoundFault
DBClusterIdentifier does not refer to an existing DB cluster.
HTTP Status Code: 404

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeDBClusterSnapshotAttributes

Returns a list of DB cluster snapshot attribute names and values for a manual DB cluster snapshot.

When sharing snapshots with other AWS accounts, DescribeDBClusterSnapshotAttributes returns the `restore` attribute and a list of IDs for the AWS accounts that are authorized to copy or restore the manual DB cluster snapshot. If `all` is included in the list of values for the `restore` attribute, then the manual DB cluster snapshot is public and can be copied or restored by all AWS accounts.

To add or remove access for an AWS account to copy or restore a manual DB cluster snapshot, or to make the manual DB cluster snapshot public or private, use the ModifyDBClusterSnapshotAttribute (p. 257) API action.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

**DBClusterSnapshotIdentifier**

The identifier for the DB cluster snapshot to describe the attributes for.

- Type: String
- Required: Yes

Response Elements

The following element is returned by the service.

**DBClusterSnapshotAttributesResult**

Contains the results of a successful call to the DescribeDBClusterSnapshotAttributes (p. 212) API action.

Manual DB cluster snapshot attributes are used to authorize other AWS accounts to copy or restore a manual DB cluster snapshot. For more information, see the ModifyDBClusterSnapshotAttribute (p. 257) API action.

- Type: DBClusterSnapshotAttributesResult (p. 317) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

**DBClusterSnapshotNotFoundFault**

- *DBClusterSnapshotIdentifier* does not refer to an existing DB cluster snapshot.
  - HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeDBClusterSnapshots

Returns information about DB cluster snapshots. This API action supports pagination.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

**DBClusterIdentifier**

The ID of the DB cluster to retrieve the list of DB cluster snapshots for. This parameter can't be used in conjunction with the `DBClusterSnapshotIdentifier` parameter. This parameter is not case-sensitive.

Constraints:

- If supplied, must match the identifier of an existing DBCluster.

Type: String

Required: No

**DBClusterSnapshotIdentifier**

A specific DB cluster snapshot identifier to describe. This parameter can't be used in conjunction with the `DBClusterIdentifier` parameter. This value is stored as a lowercase string.

Constraints:

- If supplied, must match the identifier of an existing DBClusterSnapshot.
- If this identifier is for an automated snapshot, the `SnapshotType` parameter must also be specified.

Type: String

Required: No

**Filters.Filter.N**

This parameter is not currently supported.

Type: Array of Filter (p. 342) objects

Required: No

**IncludePublic**

True to include manual DB cluster snapshots that are public and can be copied or restored by any AWS account, and otherwise false. The default is `false`. The default is `false`.

You can share a manual DB cluster snapshot as public by using the `ModifyDBClusterSnapshotAttribute (p. 257)` API action.

Type: Boolean

Required: No

**IncludeShared**

True to include shared manual DB cluster snapshots from other AWS accounts that this AWS account has been given permission to copy or restore, and otherwise false. The default is `false`.

Type: Boolean

Required: No

API Version 2017-11-29
You can give an AWS account permission to restore a manual DB cluster snapshot from another AWS account by the `ModifyDBClusterSnapshotAttribute` (p. 257) API action.

**Type:** Boolean  
**Required:** No

**Marker**
An optional pagination token provided by a previous `DescribeDBClusterSnapshots` request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by `MaxRecords`.

**Type:** String  
**Required:** No

**MaxRecords**
The maximum number of records to include in the response. If more records exist than the specified `MaxRecords` value, a pagination token called a marker is included in the response so that the remaining results can be retrieved.

**Default:** 100  
**Constraints:** Minimum 20, maximum 100.

**Type:** Integer  
**Required:** No

**SnapshotType**
The type of DB cluster snapshots to be returned. You can specify one of the following values:

- `automated` - Return all DB cluster snapshots that have been automatically taken by Amazon Neptune for my AWS account.
- `manual` - Return all DB cluster snapshots that have been taken by my AWS account.
- `shared` - Return all manual DB cluster snapshots that have been shared to my AWS account.
- `public` - Return all DB cluster snapshots that have been marked as public.

If you don't specify a `SnapshotType` value, then both automated and manual DB cluster snapshots are returned. You can include shared DB cluster snapshots with these results by setting the `IncludeShared` parameter to `true`. You can include public DB cluster snapshots with these results by setting the `IncludePublic` parameter to `true`.

The `IncludeShared` and `IncludePublic` parameters don't apply for `SnapshotType` values of `manual` or `automated`. The `IncludePublic` parameter doesn't apply when `SnapshotType` is set to `shared`. The `IncludeShared` parameter doesn't apply when `SnapshotType` is set to `public`.

**Type:** String  
**Required:** No

**Response Elements**
The following elements are returned by the service.

**`DBClusterSnapshots.DBClusterSnapshot.N`**
Provides a list of DB cluster snapshots for the user.
Type: Array of DBClusterSnapshot (p. 313) objects

Marker

An optional pagination token provided by a previous DescribeDBClusterSnapshots (p. 214) request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by MaxRecords.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

DBClusterSnapshotNotFoundFault

DBClusterSnapshotIdentifier does not refer to an existing DB cluster snapshot.

HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeDBEngineVersions

Returns a list of the available DB engines.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

**DBParameterGroupFamily**

The name of a specific DB parameter group family to return details for.

Constraints:

- If supplied, must match an existing DBParameterGroupFamily.

Type: String

Required: No

**DefaultOnly**

Indicates that only the default version of the specified engine or engine and major version combination is returned.

Type: Boolean

Required: No

**Engine**

The database engine to return.

Type: String

Required: No

**EngineVersion**

The database engine version to return.

Example: 5.1.49

Type: String

Required: No

**Filters.Filter.N**

Not currently supported.

Type: Array of Filter (p. 342) objects

Required: No

**ListSupportedCharacterSets**

If this parameter is specified and the requested engine supports the **CharacterSet** parameter for **CreateDBInstance**, the response includes a list of supported character sets for each engine version.

Type: Boolean

Required: No
ListSupportedTimezones

If this parameter is specified and the requested engine supports the TimeZone parameter for CreateDBInstance, the response includes a list of supported time zones for each engine version.

Type: Boolean
Required: No

Marker

An optional pagination token provided by a previous request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by MaxRecords.

Type: String
Required: No

MaxRecords

The maximum number of records to include in the response. If more than the MaxRecords value is available, a pagination token called a marker is included in the response so that the following results can be retrieved.

Default: 100

Constraints: Minimum 20, maximum 100.

Type: Integer
Required: No

Response Elements

The following elements are returned by the service.

DBEngineVersions.DBEngineVersion.N

A list of DBEngineVersion elements.

Type: Array of DBEngineVersion (p. 318) objects

Marker

An optional pagination token provided by a previous request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by MaxRecords.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
• AWS SDK for C++
• AWS SDK for Go
• AWS SDK for Java
• AWS SDK for JavaScript
• AWS SDK for PHP V3
• AWS SDK for Python
• AWS SDK for Ruby V2
DescribeDBInstances

Returns information about provisioned instances. This API supports pagination.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

DBInstanceIdentifier

The user-supplied instance identifier. If this parameter is specified, information from only the specific DB instance is returned. This parameter isn't case-sensitive.

Constraints:
- If supplied, must match the identifier of an existing DBInstance.

Type: String
Required: No

Filters.Filter.N

A filter that specifies one or more DB instances to describe.

Supported filters:
- db-cluster-id - Accepts DB cluster identifiers and DB cluster Amazon Resource Names (ARNs). The results list will only include information about the DB instances associated with the DB clusters identified by these ARNs.
- db-instance-id - Accepts DB instance identifiers and DB instance Amazon Resource Names (ARNs). The results list will only include information about the DB instances identified by these ARNs.

Type: Array of Filter (p. 342) objects
Required: No

Marker

An optional pagination token provided by a previous DescribeDBInstances request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by MaxRecords.

Type: String
Required: No

MaxRecords

The maximum number of records to include in the response. If more records exist than the specified MaxRecords value, a pagination token called a marker is included in the response so that the remaining results can be retrieved.

Default: 100
Constraints: Minimum 20, maximum 100.
Type: Integer
Required: No
Response Elements

The following elements are returned by the service.

**DBInstances**.DBInstance.N

A list of DBInstance (p. 320) instances.

Type: Array of DBInstance (p. 320) objects

**Marker**

An optional pagination token provided by a previous request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by MaxRecords.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

**DBInstanceNotFound**

  *DBInstanceIdentifier* does not refer to an existing DB instance.

  HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeDBParameterGroups

Returns a list of DBParameterGroup descriptions. If a DBParameterGroupName is specified, the list will contain only the description of the specified DB parameter group.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

DBParameterGroupName

The name of a specific DB parameter group to return details for.

Constraints:

- If supplied, must match the name of an existing DBClusterParameterGroup.

Type: String

Required: No

Filters.Filter.N

This parameter is not currently supported.

Type: Array of Filter (p. 342) objects

Required: No

Marker

An optional pagination token provided by a previous DescribeDBParameterGroups request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by MaxRecords.

Type: String

Required: No

MaxRecords

The maximum number of records to include in the response. If more records exist than the specified MaxRecords value, a pagination token called a marker is included in the response so that the remaining results can be retrieved.

Default: 100

Constraints: Minimum 20, maximum 100.

Type: Integer

Required: No

Response Elements

The following elements are returned by the service.

DBParameterGroups.DBParameterGroup.N

A list of DBParameterGroup (p. 328) instances.
Type: Array of `DBParameterGroup` (p. 328) objects

**Marker**

An optional pagination token provided by a previous request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by `MaxRecords`.

Type: String

**Errors**

For information about the errors that are common to all actions, see Common Errors (p. 363).

**DBParameterGroupNotFound**

`DBParameterGroupName` does not refer to an existing DB parameter group.

HTTP Status Code: 404

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeDBParameters

Returns the detailed parameter list for a particular DB parameter group.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

DBParameterGroupName

The name of a specific DB parameter group to return details for.

Constraints:
• If supplied, must match the name of an existing DBParameterGroup.

Type: String
Required: Yes

Filters.Filter.N

This parameter is not currently supported.

Type: Array of Filter (p. 342) objects
Required: No

Marker

An optional pagination token provided by a previous DescribeDBParameters request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by MaxRecords.

Type: String
Required: No

MaxRecords

The maximum number of records to include in the response. If more records exist than the specified MaxRecords value, a pagination token called a marker is included in the response so that the remaining results can be retrieved.

Default: 100
Constraints: Minimum 20, maximum 100.

Type: Integer
Required: No

Source

The parameter types to return.

Default: All parameter types returned

Valid Values: user | system | engine-default

Type: String
Required: No
Response Elements

The following elements are returned by the service.

Marker

An optional pagination token provided by a previous request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by MaxRecords.

Type: String

Parameters.Parameter.N

A list of Parameter (p. 347) values.

Type: Array of Parameter (p. 347) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

DBParameterGroupNotFound

DBParameterGroupName does not refer to an existing DB parameter group.

HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeDBSubnetGroups

Returns a list of DBSubnetGroup descriptions. If a DBSubnetGroupName is specified, the list will contain only the descriptions of the specified DBSubnetGroup.

For an overview of CIDR ranges, go to the Wikipedia Tutorial.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

DBSubnetGroupName

The name of the DB subnet group to return details for.

Type: String

Required: No

Filters.Filter.N

This parameter is not currently supported.

Type: Array of Filter (p. 342) objects

Required: No

Marker

An optional pagination token provided by a previous DescribeDBSubnetGroups request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by MaxRecords.

Type: String

Required: No

MaxRecords

The maximum number of records to include in the response. If more records exist than the specified MaxRecords value, a pagination token called a marker is included in the response so that the remaining results can be retrieved.

Default: 100

Constraints: Minimum 20, maximum 100.

Type: Integer

Required: No

Response Elements

The following elements are returned by the service.

DBSubnetGroups.DBSubnetGroup.N

A list of DBSubnetGroup (p. 331) instances.

Type: Array of DBSubnetGroup (p. 331) objects
Marker

An optional pagination token provided by a previous request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by MaxRecords.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

DBSubnetGroupNotFoundFault

*DBSubnetGroupName* does not refer to an existing DB subnet group.

HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeEngineDefaultClusterParameters

Returns the default engine and system parameter information for the cluster database engine.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

DBParameterGroupFamily

The name of the DB cluster parameter group family to return engine parameter information for.

Type: String

Required: Yes

Filters.Filter.N

This parameter is not currently supported.

Type: Array of Filter (p. 342) objects

Required: No

Marker

An optional pagination token provided by a previous DescribeEngineDefaultClusterParameters request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by MaxRecords.

Type: String

Required: No

MaxRecords

The maximum number of records to include in the response. If more records exist than the specified MaxRecords value, a pagination token called a marker is included in the response so that the remaining results can be retrieved.

Default: 100

Constraints: Minimum 20, maximum 100.

Type: Integer

Required: No

Response Elements

The following element is returned by the service.

EngineDefaults

Contains the result of a successful invocation of the DescribeEngineDefaultParameters (p. 230) action.

Type: EngineDefaults (p. 336) object
Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeEngineDefaultParameters

Returns the default engine and system parameter information for the specified database engine.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

DBParameterGroupFamily

The name of the DB parameter group family.

Type: String

Required: Yes

Filters.Filter.N

Not currently supported.

Type: Array of Filter (p. 342) objects

Required: No

Marker

An optional pagination token provided by a previous DescribeEngineDefaultParameters request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by MaxRecords.

Type: String

Required: No

MaxRecords

The maximum number of records to include in the response. If more records exist than the specified MaxRecords value, a pagination token called a marker is included in the response so that the remaining results can be retrieved.

Default: 100

Constraints: Minimum 20, maximum 100.

Type: Integer

Required: No

Response Elements

The following element is returned by the service.

EngineDefaults

Contains the result of a successful invocation of the DescribeEngineDefaultParameters (p. 230) action.

Type: EngineDefaults (p. 336) object
Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeEventCategories

Displays a list of categories for all event source types, or, if specified, for a specified source type.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

Filters.Filter.N

This parameter is not currently supported.

Type: Array of Filter (p. 342) objects
Required: No

SourceType

The type of source that is generating the events.

Valid values: db-instance | db-parameter-group | db-security-group | db-snapshot

Type: String
Required: No

Response Elements

The following element is returned by the service.

EventCategoriesMapList.EventCategoriesMap.N

A list of EventCategoriesMap data types.

Type: Array of EventCategoriesMap (p. 339) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeEvents

Returns events related to DB instances, DB security groups, DB snapshots, and DB parameter groups for the past 14 days. Events specific to a particular DB instance, DB security group, database snapshot, or DB parameter group can be obtained by providing the name as a parameter. By default, the past hour of events are returned.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

Duration

The number of minutes to retrieve events for.

Default: 60

Type: Integer

Required: No

EndTime

The end of the time interval for which to retrieve events, specified in ISO 8601 format. For more information about ISO 8601, go to the ISO8601 Wikipedia page.

Example: 2009-07-08T18:00Z

Type: Timestamp

Required: No

EventCategories.EventCategory.N

A list of event categories that trigger notifications for an event notification subscription.

Type: Array of strings

Required: No

Filters.Filter.N

This parameter is not currently supported.

Type: Array of Filter (p. 342) objects

Required: No

Marker

An optional pagination token provided by a previous DescribeEvents request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by MaxRecords.

Type: String

Required: No

MaxRecords

The maximum number of records to include in the response. If more records exist than the specified MaxRecords value, a pagination token called a marker is included in the response so that the remaining results can be retrieved.
DescribeEvents

Default: 100
Constraints: Minimum 20, maximum 100.
Type: Integer
Required: No

SourceIdentifier

The identifier of the event source for which events are returned. If not specified, then all sources are included in the response.

Constraints:
- If SourceIdentifier is supplied, SourceType must also be provided.
- If the source type is DBInstance, then a DBInstanceIdentifier must be supplied.
- If the source type is DBSecurityGroup, a DBSecurityGroupName must be supplied.
- If the source type is DBParameterGroup, a DBParameterGroupName must be supplied.
- If the source type is DBSnapshot, a DBSnapshotIdentifier must be supplied.
- Cannot end with a hyphen or contain two consecutive hyphens.

Type: String
Required: No

SourceType

The event source to retrieve events for. If no value is specified, all events are returned.

Type: String

Valid Values: db-instance | db-parameter-group | db-security-group | db-snapshot | db-cluster | db-cluster-snapshot

Required: No

StartTime

The beginning of the time interval to retrieve events for, specified in ISO 8601 format. For more information about ISO 8601, go to the ISO8601 Wikipedia page.

Example: 2009-07-08T18:00Z

Type: Timestamp
Required: No

Response Elements

The following elements are returned by the service.

Events.Event.N

A list of Event (p. 337) instances.

Type: Array of Event (p. 337) objects

Marker

An optional pagination token provided by a previous Events request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by MaxRecords.
Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeEventSubscriptions

Lists all the subscription descriptions for a customer account. The description for a subscription includes SubscriptionName, SNSTopicARN, CustomerID, SourceType, SourceID, CreationTime, and Status.

If you specify a SubscriptionName, lists the description for that subscription.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

Filters.Filter.N

This parameter is not currently supported.

Type: Array of Filter (p. 342) objects

Required: No

Marker

An optional pagination token provided by a previous DescribeOrderableDBInstanceOptions request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by MaxRecords.

Type: String

Required: No

MaxRecords

The maximum number of records to include in the response. If more records exist than the specified MaxRecords value, a pagination token called a marker is included in the response so that the remaining results can be retrieved.

Default: 100

Constraints: Minimum 20, maximum 100.

Type: Integer

Required: No

SubscriptionName

The name of the event notification subscription you want to describe.

Type: String

Required: No

Response Elements

The following elements are returned by the service.

EventSubscriptionsList.EventSubscription.N

A list of EventSubscriptions data types.

Type: Array of EventSubscription (p. 340) objects
Marker

An optional pagination token provided by a previous DescribeOrderableDBInstanceOptions request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by MaxRecords.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

SubscriptionNotFoundException

HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeOrderableDBInstanceOptions

Returns a list of orderable DB instance options for the specified engine.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

**DBInstanceClass**

The DB instance class filter value. Specify this parameter to show only the available offerings matching the specified DB instance class.

Type: String

Required: No

**Engine**

The name of the engine to retrieve DB instance options for.

Type: String

Required: Yes

**EngineVersion**

The engine version filter value. Specify this parameter to show only the available offerings matching the specified engine version.

Type: String

Required: No

**Filters.Filter.N**

This parameter is not currently supported.

Type: Array of Filter (p. 342) objects

Required: No

**LicenseModel**

The license model filter value. Specify this parameter to show only the available offerings matching the specified license model.

Type: String

Required: No

**Marker**

An optional pagination token provided by a previous DescribeOrderableDBInstanceOptions request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by MaxRecords.

Type: String

Required: No
**MaxRecords**

The maximum number of records to include in the response. If more records exist than the specified `MaxRecords` value, a pagination token called a marker is included in the response so that the remaining results can be retrieved.

Default: 100
Constraints: Minimum 20, maximum 100.
Type: Integer
Required: No

**Vpc**

The VPC filter value. Specify this parameter to show only the available VPC or non-VPC offerings.

Type: Boolean
Required: No

### Response Elements

The following elements are returned by the service.

**Marker**

An optional pagination token provided by a previous `DescribeOrderableDBInstanceOptions` request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by `MaxRecords`.

Type: String

**OrderableDBInstanceOptions.OrderableDBInstanceOption.N**

An `OrderableDBInstanceOption` (p. 344) structure containing information about orderable options for the DB instance.

Type: Array of `OrderableDBInstanceOption` (p. 344) objects

### Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

### See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
• AWS SDK for Ruby V2
DescribePendingMaintenanceActions

Returns a list of resources (for example, DB instances) that have at least one pending maintenance action.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

Filters.Filter.N

A filter that specifies one or more resources to return pending maintenance actions for.

Supported filters:
- `db-cluster-id` - Accepts DB cluster identifiers and DB cluster Amazon Resource Names (ARNs). The results list will only include pending maintenance actions for the DB clusters identified by these ARNs.
- `db-instance-id` - Accepts DB instance identifiers and DB instance ARNs. The results list will only include pending maintenance actions for the DB instances identified by these ARNs.

Type: Array of Filter (p. 342) objects

Required: No

Marker

An optional pagination token provided by a previous DescribePendingMaintenanceActions request. If this parameter is specified, the response includes only records beyond the marker, up to a number of records specified by MaxRecords.

Type: String

Required: No

MaxRecords

The maximum number of records to include in the response. If more records exist than the specified MaxRecords value, a pagination token called a marker is included in the response so that the remaining results can be retrieved.

Default: 100

Constraints: Minimum 20, maximum 100.

Type: Integer

Required: No

ResourceIdentifier

The ARN of a resource to return pending maintenance actions for.

Type: String

Required: No

Response Elements

The following elements are returned by the service.
Marker

An optional pagination token provided by a previous DescribePendingMaintenanceActions request. If this parameter is specified, the response includes only records beyond the marker, up to a number of records specified by MaxRecords.

Type: String

PendingMaintenanceActions.ResourcePendingMaintenanceActions.N

A list of the pending maintenance actions for the resource.

Type: Array of ResourcePendingMaintenanceActions (p. 356) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

ResourceNotFoundException

The specified resource ID was not found.

HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeValidDBInstanceModifications

You can call DescribeValidDBInstanceModifications (p. 244) to learn what modifications you can make to your DB instance. You can use this information when you call ModifyDBInstance (p. 260).

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

DBInstanceIdentifier

The customer identifier or the ARN of your DB instance.

Type: String

Required: Yes

Response Elements

The following element is returned by the service.

ValidDBInstanceModificationsMessage

Information about valid modifications that you can make to your DB instance. Contains the result of a successful call to the DescribeValidDBInstanceModifications (p. 244) action. You can use this information when you call ModifyDBInstance (p. 260).

Type: ValidDBInstanceModificationsMessage (p. 361) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

DBInstanceNotFound

DBInstanceIdentifier does not refer to an existing DB instance.

HTTP Status Code: 404

InvalidDBInstanceState

The specified DB instance is not in the available state.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
• AWS SDK for JavaScript
• AWS SDK for PHP V3
• AWS SDK for Python
• AWS SDK for Ruby V2
FailoverDBCluster

Forces a failover for a DB cluster.

A failover for a DB cluster promotes one of the Read Replicas (read-only instances) in the DB cluster to be
the primary instance (the cluster writer).

Amazon Neptune will automatically fail over to a Read Replica, if one exists, when the primary instance
fails. You can force a failover when you want to simulate a failure of a primary instance for testing.
Because each instance in a DB cluster has its own endpoint address, you will need to clean up and re-
establish any existing connections that use those endpoint addresses when the failover is complete.

Request Parameters

For information about the parameters that are common to all actions, see Common
Parameters (p. 365).

DBClusterIdentifier

A DB cluster identifier to force a failover for. This parameter is not case-sensitive.

Constraints:
- Must match the identifier of an existing DBCluster.

Type: String

Required: No

TargetDBInstanceIdentifier

The name of the instance to promote to the primary instance.

You must specify the instance identifier for an Read Replica in the DB cluster. For example,
mydbcluster-replica1.

Type: String

Required: No

Response Elements

The following element is returned by the service.

DBCluster

Contains the details of an Amazon Neptune DB cluster.

This data type is used as a response element in the DescribeDBClusters (p. 210) action.

Type: DBCluster (p. 304) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

DBClusterNotFoundFault

DBClusterIdentifier does not refer to an existing DB cluster.
HTTP Status Code: 404

InvalidDBClusterStateFault

The DB cluster is not in a valid state.

HTTP Status Code: 400

InvalidDBInstanceState

The specified DB instance is not in the available state.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
ListTagsForResource

Lists all tags on an Amazon Neptune resource.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

Filters.Filter.N

This parameter is not currently supported.

Type: Array of Filter (p. 342) objects

Required: No

ResourceName

The Amazon Neptune resource with tags to be listed. This value is an Amazon Resource Name (ARN).

For information about creating an ARN, see Constructing an Amazon Resource Name (ARN).

Type: String

Required: Yes

Response Elements

The following element is returned by the service.

TagList.Tag.N

List of tags returned by the ListTagsForResource operation.

Type: Array of Tag (p. 358) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

DBClusterNotFoundFault

DBClusterIdentifier does not refer to an existing DB cluster.

HTTP Status Code: 404

DBInstanceNotFound

DBInstanceIdentifier does not refer to an existing DB instance.

HTTP Status Code: 404

DBSnapshotNotFound

DBSnapshotIdentifier does not refer to an existing DB snapshot.

HTTP Status Code: 404
See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
ModifyDBCluster

Modify a setting for a DB cluster. You can change one or more database configuration parameters by specifying these parameters and the new values in the request.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

**ApplyImmediately**

A value that specifies whether the modifications in this request and any pending modifications are asynchronously applied as soon as possible, regardless of the PreferredMaintenanceWindow setting for the DB cluster. If this parameter is set to false, changes to the DB cluster are applied during the next maintenance window.

The ApplyImmediately parameter only affects the NewDBClusterIdentifier and MasterUserPassword values. If you set the ApplyImmediately parameter value to false, then changes to the NewDBClusterIdentifier and MasterUserPassword values are applied during the next maintenance window. All other changes are applied immediately, regardless of the value of the ApplyImmediately parameter.

Default: false

Type: Boolean

Required: No

**BackupRetentionPeriod**

The number of days for which automated backups are retained. You must specify a minimum value of 1.

Default: 1

Constraints:

- Must be a value from 1 to 35

Type: Integer

Required: No

**DBClusterIdentifier**

The DB cluster identifier for the cluster being modified. This parameter is not case-sensitive.

Constraints:

- Must match the identifier of an existing DBCluster.

Type: String

Required: Yes

**DBClusterParameterGroupName**

The name of the DB cluster parameter group to use for the DB cluster.

Type: String

Required: No
EnableIAMDatabaseAuthentication

True to enable mapping of AWS Identity and Access Management (IAM) accounts to database accounts, and otherwise false.

Default: false

Type: Boolean

Required: No

EngineVersion

The version number of the database engine to which you want to upgrade. Changing this parameter results in an outage. The change is applied during the next maintenance window unless the ApplyImmediately parameter is set to true.

For a list of valid engine versions, see CreateDBInstance (p. 175), or call DescribeDBEngineVersions (p. 217).

Type: String

Required: No

MasterUserPassword

The new password for the master database user. This password can contain any printable ASCII character except "/", """, or ".@

Constraints: Must contain from 8 to 41 characters.

Type: String

Required: No

NewDBClusterIdentifier

The new DB cluster identifier for the DB cluster when renaming a DB cluster. This value is stored as a lowercase string.

Constraints:
- Must contain from 1 to 63 letters, numbers, or hyphens
- The first character must be a letter
- Cannot end with a hyphen or contain two consecutive hyphens

Example: my-cluster2

Type: String

Required: No

OptionGroupName

A value that indicates that the DB cluster should be associated with the specified option group. Changing this parameter doesn't result in an outage except in the following case, and the change is applied during the next maintenance window unless the ApplyImmediately parameter is set to true for this request. If the parameter change results in an option group that enables OEM, this change can cause a brief (sub-second) period during which new connections are rejected but existing connections are not interrupted.

Permanent options can't be removed from an option group. The option group can't be removed from a DB cluster once it is associated with a DB cluster.

Type: String
Required: No

**Port**

The port number on which the DB cluster accepts connections.

Constraints: Value must be 1150-65535

Default: The same port as the original DB cluster.

Type: Integer

Required: No

**PreferredBackupWindow**

The daily time range during which automated backups are created if automated backups are enabled, using the **BackupRetentionPeriod** parameter.

The default is a 30-minute window selected at random from an 8-hour block of time for each AWS Region.

Constraints:
- Must be in the format hh24:mi-hh24:mi.
- Must be in Universal Coordinated Time (UTC).
- Must not conflict with the preferred maintenance window.
- Must be at least 30 minutes.

Type: String

Required: No

**PreferredMaintenanceWindow**

The weekly time range during which system maintenance can occur, in Universal Coordinated Time (UTC).


The default is a 30-minute window selected at random from an 8-hour block of time for each AWS Region, occurring on a random day of the week.

Valid Days: Mon, Tue, Wed, Thu, Fri, Sat, Sun.

Constraints: Minimum 30-minute window.

Type: String

Required: No

**VpcSecurityGroupIds.VpcSecurityGroupId.N**

A list of VPC security groups that the DB cluster will belong to.

Type: Array of strings

Required: No

---

**Response Elements**

The following element is returned by the service.
**DBCluster**

Contains the details of an Amazon Neptune DB cluster.

This data type is used as a response element in the **DescribeDBClusters (p. 210)** action.

Type: **DBCluster (p. 304)** object

**Errors**

For information about the errors that are common to all actions, see **Common Errors (p. 363)**.

**DBClusterAlreadyExistsFault**

User already has a DB cluster with the given identifier.

HTTP Status Code: 400  
**DBClusterNotFoundFault**

**DBClusterIdentifier** does not refer to an existing DB cluster.

HTTP Status Code: 404  
**DBClusterParameterGroupNotFound**

**DBClusterParameterGroupName** does not refer to an existing DB Cluster parameter group.

HTTP Status Code: 404  
**DBSubnetGroupNotFoundFault**

**DBSubnetGroupName** does not refer to an existing DB subnet group.

HTTP Status Code: 404  
**InvalidDBClusterStateFault**

The DB cluster is not in a valid state.

HTTP Status Code: 400  
**InvalidDBInstanceState**

The specified DB instance is not in the **available** state.

HTTP Status Code: 400  
**InvalidDBSecurityGroupState**

The state of the DB security group does not allow deletion.

HTTP Status Code: 400  
**InvalidDBSubnetGroupStateFault**

The DB subnet group cannot be deleted because it is in use.

HTTP Status Code: 400  
**InvalidSubnet**

The requested subnet is invalid, or multiple subnets were requested that are not all in a common VPC.

HTTP Status Code: 400
InvalidVPCNetworkStateFault

DB subnet group does not cover all Availability Zones after it is created because users' change.

HTTP Status Code: 400

StorageQuotaExceeded

Request would result in user exceeding the allowed amount of storage available across all DB instances.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
ModifyDBClusterParameterGroup

Modifies the parameters of a DB cluster parameter group. To modify more than one parameter, submit a list of the following: ParameterName, ParameterValue, and ApplyMethod. A maximum of 20 parameters can be modified in a single request.

**Note**
Changes to dynamic parameters are applied immediately. Changes to static parameters require a reboot without failover to the DB cluster associated with the parameter group before the change can take effect.

**Important**
After you create a DB cluster parameter group, you should wait at least 5 minutes before creating your first DB cluster that uses that DB cluster parameter group as the default parameter group. This allows Amazon Neptune to fully complete the create action before the parameter group is used as the default for a new DB cluster. This is especially important for parameters that are critical when creating the default database for a DB cluster, such as the character set for the default database defined by the `character_set_database` parameter. You can use the Parameter Groups option of the Amazon Neptune console or the DescribeDBClusterParameters (p. 208) command to verify that your DB cluster parameter group has been created or modified.

**Request Parameters**

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

**DBClusterParameterGroupName**

The name of the DB cluster parameter group to modify.

Type: String

Required: Yes

**Parameters.Parameter.N**

A list of parameters in the DB cluster parameter group to modify.

Type: Array of Parameter (p. 347) objects

Required: Yes

**Response Elements**

The following element is returned by the service.

**DBClusterParameterGroupName**

The name of the DB cluster parameter group.

Constraints:
- Must be 1 to 255 letters or numbers.
- First character must be a letter
- Cannot end with a hyphen or contain two consecutive hyphens

**Note**
This value is stored as a lowercase string.
Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

DBParameterGroupNameNotFound

*DBParameterGroupName* does not refer to an existing DB parameter group.

HTTP Status Code: 404

InvalidDBParameterGroupState

The DB parameter group is in use or is in an invalid state. If you are attempting to delete the parameter group, you cannot delete it when the parameter group is in this state.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
ModifyDBClusterSnapshotAttribute

Adds an attribute and values to, or removes an attribute and values from, a manual DB cluster snapshot.

To share a manual DB cluster snapshot with other AWS accounts, specify restore as the AttributeName and use the ValuesToAdd parameter to add a list of IDs of the AWS accounts that are authorized to restore the manual DB cluster snapshot. Use the value all to make the manual DB cluster snapshot public, which means that it can be copied or restored by all AWS accounts. Do not add the all value for any manual DB cluster snapshots that contain private information that you don't want available to all AWS accounts. If a manual DB cluster snapshot is encrypted, it can be shared, but only by specifying a list of authorized AWS account IDs for the ValuesToAdd parameter. You can't use all as a value for that parameter in this case.

To view which AWS accounts have access to copy or restore a manual DB cluster snapshot, or whether a manual DB cluster snapshot public or private, use the DescribeDBClusterSnapshotAttributes API action.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters.

AttributeName

The name of the DB cluster snapshot attribute to modify.

To manage authorization for other AWS accounts to copy or restore a manual DB cluster snapshot, set this value to restore.

Type: String

Required: Yes

DBClusterSnapshotIdentifier

The identifier for the DB cluster snapshot to modify the attributes for.

Type: String

Required: Yes

ValuesToAdd.AttributeValue.N

A list of DB cluster snapshot attributes to add to the attribute specified by AttributeName.

To authorize other AWS accounts to copy or restore a manual DB cluster snapshot, set this list to include one or more AWS account IDs, or all to make the manual DB cluster snapshot restorable by any AWS account. Do not add the all value for any manual DB cluster snapshots that contain private information that you don't want available to all AWS accounts.

Type: Array of strings

Required: No

ValuesToRemove.AttributeValue.N

A list of DB cluster snapshot attributes to remove from the attribute specified by AttributeName.

To remove authorization for other AWS accounts to copy or restore a manual DB cluster snapshot, set this list to include one or more AWS account identifiers, or all to remove authorization for any AWS account to copy or restore the DB cluster snapshot. If you specify all, an AWS account whose...
account ID is explicitly added to the `restore` attribute can still copy or restore a manual DB cluster snapshot.

Type: Array of strings

Required: No

**Response Elements**

The following element is returned by the service.

**DBClusterSnapshotAttributesResult**

Contains the results of a successful call to the DescribeDBClusterSnapshotAttributes (p. 212) API action.

Manual DB cluster snapshot attributes are used to authorize other AWS accounts to copy or restore a manual DB cluster snapshot. For more information, see the ModifyDBClusterSnapshotAttribute (p. 257) API action.

Type: `DBClusterSnapshotAttributesResult` (p. 317) object

**Errors**

For information about the errors that are common to all actions, see Common Errors (p. 363).

**DBClusterSnapshotNotFoundFault**

`DBClusterSnapshotIdentifier` does not refer to an existing DB cluster snapshot.

HTTP Status Code: 404

**InvalidDBClusterSnapshotStateFault**

The supplied value is not a valid DB cluster snapshot state.

HTTP Status Code: 400

**SharedSnapshotQuotaExceeded**

You have exceeded the maximum number of accounts that you can share a manual DB snapshot with.

HTTP Status Code: 400

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
• AWS SDK for Python
• AWS SDK for Ruby V2
ModifyDBInstance

Modifies settings for a DB instance. You can change one or more database configuration parameters by specifying these parameters and the new values in the request. To learn what modifications you can make to your DB instance, call DescribeValidDBInstanceModifications (p. 244) before you call ModifyDBInstance (p. 260).

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

AllocatedStorage

The new amount of storage (in gibibytes) to allocate for the DB instance.

Not applicable. Storage is managed by the DB Cluster.

Type: Integer

Required: No

AllowMajorVersionUpgrade

Indicates that major version upgrades are allowed. Changing this parameter doesn't result in an outage and the change is asynchronously applied as soon as possible.

Constraints: This parameter must be set to true when specifying a value for the EngineVersion parameter that is a different major version than the DB instance's current version.

Type: Boolean

Required: No

ApplyImmediately

Specifies whether the modifications in this request and any pending modifications are asynchronously applied as soon as possible, regardless of the PreferredMaintenanceWindow setting for the DB instance.

If this parameter is set to false, changes to the DB instance are applied during the next maintenance window. Some parameter changes can cause an outage and are applied on the next call to RebootDBInstance (p. 278), or the next failure reboot.

Default: false

Type: Boolean

Required: No

AutoMinorVersionUpgrade

Indicates that minor version upgrades are applied automatically to the DB instance during the maintenance window. Changing this parameter doesn't result in an outage except in the following case and the change is asynchronously applied as soon as possible. An outage will result if this parameter is set to true during the maintenance window, and a newer minor version is available, and Neptune has enabled auto patching for that engine version.

Type: Boolean

Required: No
BackupRetentionPeriod

The number of days to retain automated backups. Setting this parameter to a positive number enables backups. Setting this parameter to 0 disables automated backups.

Not applicable. The retention period for automated backups is managed by the DB cluster. For more information, see ModifyDBCluster (p. 250).

Default: Uses existing setting

Type: Integer

Required: No

CACertificateIdentifier

Indicates the certificate that needs to be associated with the instance.

Type: String

Required: No

CloudwatchLogsExportConfiguration

The configuration setting for the log types to be enabled for export to CloudWatch Logs for a specific DB instance or DB cluster.

Type: CloudwatchLogsExportConfiguration (p. 303) object

Required: No

CopyTagsToSnapshot

True to copy all tags from the DB instance to snapshots of the DB instance, and otherwise false. The default is false.

Type: Boolean

Required: No

DBInstanceClass

The new compute and memory capacity of the DB instance, for example, db.m4.large. Not all DB instance classes are available in all AWS Regions.

If you modify the DB instance class, an outage occurs during the change. The change is applied during the next maintenance window, unless ApplyImmediately is specified as true for this request.

Default: Uses existing setting

Type: String

Required: No

DBInstanceIdentifier

The DB instance identifier. This value is stored as a lowercase string.

Constraints:
• Must match the identifier of an existing DBInstance.

Type: String

Required: Yes
**DBParameterGroupName**

The name of the DB parameter group to apply to the DB instance. Changing this setting doesn't result in an outage. The parameter group name itself is changed immediately, but the actual parameter changes are not applied until you reboot the instance without failover. The db instance will NOT be rebooted automatically and the parameter changes will NOT be applied during the next maintenance window.

Default: Uses existing setting

Constraints: The DB parameter group must be in the same DB parameter group family as this DB instance.

Type: String

Required: No

**DBPortNumber**

The port number on which the database accepts connections.

The value of the `DBPortNumber` parameter must not match any of the port values specified for options in the option group for the DB instance.

Your database will restart when you change the `DBPortNumber` value regardless of the value of the `ApplyImmediately` parameter.

Default: 8182

Type: Integer

Required: No

**DBSecurityGroups.DBSecurityGroupName.N**

A list of DB security groups to authorize on this DB instance. Changing this setting doesn't result in an outage and the change is asynchronously applied as soon as possible.

Constraints:
- If supplied, must match existing DBSecurityGroups.

Type: Array of strings

Required: No

**DBSubnetGroupName**

The new DB subnet group for the DB instance. You can use this parameter to move your DB instance to a different VPC.

Changing the subnet group causes an outage during the change. The change is applied during the next maintenance window, unless you specify `true` for the `ApplyImmediately` parameter.

Constraints: If supplied, must match the name of an existing DBSubnetGroup.

Example: `mySubnetGroup`

Type: String

Required: No

**Domain**

Not supported.
Type: String
Required: No

**DomainIAMRoleName**

Not supported
Type: String
Required: No

**EnableIAMDatabaseAuthentication**

True to enable mapping of AWS Identity and Access Management (IAM) accounts to database accounts, and otherwise false.

You can enable IAM database authentication for the following database engines

Not applicable. Mapping AWS IAM accounts to database accounts is managed by the DB cluster. For more information, see `ModifyDBCluster` (p. 250).

Default: `false`
Type: Boolean
Required: No

**EnablePerformanceInsights**

True to enable Performance Insights for the DB instance, and otherwise false.

Type: Boolean
Required: No

**EngineVersion**

The version number of the database engine to upgrade to. Changing this parameter results in an outage and the change is applied during the next maintenance window unless the `ApplyImmediately` parameter is set to `true` for this request.

For major version upgrades, if a nondefault DB parameter group is currently in use, a new DB parameter group in the DB parameter group family for the new engine version must be specified. The new DB parameter group can be the default for that DB parameter group family.

Type: String
Required: No

**Iops**

The new Provisioned IOPS (I/O operations per second) value for the instance.

Changing this setting doesn't result in an outage and the change is applied during the next maintenance window unless the `ApplyImmediately` parameter is set to `true` for this request.

Default: Uses existing setting
Type: Integer
Required: No

**LicenseModel**

The license model for the DB instance.
Valid values: license-included | bring-your-own-license | general-public-license

Type: String

Required: No

**MasterUserPassword**

The new password for the master user. The password can include any printable ASCII character except "/", ",", or ".".

Not applicable.

Default: Uses existing setting

Type: String

Required: No

**MonitoringInterval**

The interval, in seconds, between points when Enhanced Monitoring metrics are collected for the DB instance. To disable collecting Enhanced Monitoring metrics, specify 0. The default is 0.

If **MonitoringRoleArn** is specified, then you must also set **MonitoringInterval** to a value other than 0.

Valid Values: 0, 1, 5, 10, 15, 30, 60

Type: Integer

Required: No

**MonitoringRoleArn**

The ARN for the IAM role that permits Neptune to send enhanced monitoring metrics to Amazon CloudWatch Logs. For example, arn:aws:iam:123456789012:role/emaccess.

If **MonitoringInterval** is set to a value other than 0, then you must supply a **MonitoringRoleArn** value.

Type: String

Required: No

**MultiAZ**

Specifies if the DB instance is a Multi-AZ deployment. Changing this parameter doesn't result in an outage and the change is applied during the next maintenance window unless the **ApplyImmediately** parameter is set to true for this request.

Type: Boolean

Required: No

**NewDBInstanceIdentifier**

The new DB instance identifier for the DB instance when renaming a DB instance. When you change the DB instance identifier, an instance reboot will occur immediately if you set **Apply Immediately** to true, or will occur during the next maintenance window if **Apply Immediately** to false. This value is stored as a lowercase string.

Constraints:

- Must contain from 1 to 63 letters, numbers, or hyphens.
The first character must be a letter.
Cannot end with a hyphen or contain two consecutive hyphens.

Example: mydbinstance

Type: String
Required: No

OptionGroupName
Indicates that the DB instance should be associated with the specified option group. Changing this parameter doesn't result in an outage except in the following case and the change is applied during the next maintenance window unless the ApplyImmediately parameter is set to true for this request. If the parameter change results in an option group that enables OEM, this change can cause a brief (sub-second) period during which new connections are rejected but existing connections are not interrupted.

Permanent options, such as the TDE option for Oracle Advanced Security TDE, can’t be removed from an option group, and that option group can't be removed from a DB instance once it is associated with a DB instance

Type: String
Required: No

PerformanceInsightsKMSKeyId
The AWS KMS key identifier for encryption of Performance Insights data. The KMS key ID is the Amazon Resource Name (ARN), KMS key identifier, or the KMS key alias for the KMS encryption key.

Type: String
Required: No

PreferredBackupWindow
The daily time range during which automated backups are created if automated backups are enabled.

Not applicable. The daily time range for creating automated backups is managed by the DB cluster. For more information, see ModifyDBCluster (p. 250).

Constraints:
• Must be in the format hh24:mi-hh24:mi
• Must be in Universal Time Coordinated (UTC)
• Must not conflict with the preferred maintenance window
• Must be at least 30 minutes

Type: String
Required: No

PreferredMaintenanceWindow
The weekly time range (in UTC) during which system maintenance can occur, which might result in an outage. Changing this parameter doesn’t result in an outage, except in the following situation, and the change is asynchronously applied as soon as possible. If there are pending actions that cause a reboot, and the maintenance window is changed to include the current time, then changing this parameter will cause a reboot of the DB instance. If moving this window to the current time, there must be at least 30 minutes between the current time and end of the window to ensure pending changes are applied.
ModifyDBInstance

Default: Uses existing setting
Valid Days: Mon | Tue | Wed | Thu | Fri | Sat | Sun
Constraints: Must be at least 30 minutes
Type: String
Required: No

PromotionTier
A value that specifies the order in which a Read Replica is promoted to the primary instance after a
failure of the existing primary instance.
Default: 1
Valid Values: 0 - 15
Type: Integer
Required: No

PubliclyAccessible
This parameter has been deprecated.
This parameter is not supported.
Type: Boolean
Required: No

StorageType
Specifies the storage type to be associated with the DB instance.
If you specify Provisioned IOPS (io1), you must also include a value for the Iops parameter.
If you choose to migrate your DB instance from using standard storage to using Provisioned IOPS,
or from using Provisioned IOPS to using standard storage, the process can take time. The duration
of the migration depends on several factors such as database load, storage size, storage type
(standard or Provisioned IOPS), amount of IOPS provisioned (if any), and the number of prior scale
storage operations. Typical migration times are under 24 hours, but the process can take up to
several days in some cases. During the migration, the DB instance is available for use, but might
experience performance degradation. While the migration takes place, nightly backups for the
instance are suspended. No other Amazon Neptune operations can take place for the instance,
including modifying the instance, rebooting the instance, deleting the instance, creating a Read
Replica for the instance, and creating a DB snapshot of the instance.
Valid values: standard | gp2 | io1
Default: io1 if the Iops parameter is specified, otherwise standard
Type: String
Required: No

TdeCredentialArn
The ARN from the key store with which to associate the instance for TDE encryption.
Type: String
Required: No

**TdeCredentialPassword**

The password for the given ARN from the key store in order to access the device.

Type: String
Required: No

**VpcSecurityGroupIds.VpcSecurityGroupId.N**

A list of EC2 VPC security groups to authorize on this DB instance. This change is asynchronously applied as soon as possible.

Not applicable. The associated list of EC2 VPC security groups is managed by the DB cluster. For more information, see [ModifyDBCluster](p. 250).

Constraints:
- If supplied, must match existing VpcSecurityGroupIds.

Type: Array of strings
Required: No

### Response Elements

The following element is returned by the service.

**DBInstance**

Contains the details of an Amazon Neptune DB instance.

This data type is used as a response element in the [DescribeDBInstances](p. 220) action.

Type: [DBInstance](p. 320) object

### Errors

For information about the errors that are common to all actions, see [Common Errors](p. 363).

#### AuthorizationNotFound

Specified CIDRIP or EC2 security group is not authorized for the specified DB security group.

Neptune may not also be authorized via IAM to perform necessary actions on your behalf.

HTTP Status Code: 404

#### CertificateNotFound

*CertificateIdentifier* does not refer to an existing certificate.

HTTP Status Code: 404

#### DBInstanceAlreadyExists

User already has a DB instance with the given identifier.

HTTP Status Code: 400
DBInstanceNotFound

DBInstanceIdentifier does not refer to an existing DB instance.

HTTP Status Code: 404

DBParameterGroupNotFound

DBParameterGroupName does not refer to an existing DB parameter group.

HTTP Status Code: 404

DBSecurityGroupNotFound

DBSecurityGroupName does not refer to an existing DB security group.

HTTP Status Code: 404

DBUpgradeDependencyFailure

The DB upgrade failed because a resource the DB depends on could not be modified.

HTTP Status Code: 400

DomainNotFoundFault

Domain does not refer to an existing Active Directory Domain.

HTTP Status Code: 404

InsufficientDBInstanceCapacity

Specified DB instance class is not available in the specified Availability Zone.

HTTP Status Code: 400

InvalidDBInstanceState

The specified DB instance is not in the available state.

HTTP Status Code: 400

InvalidDBSecurityGroupState

The state of the DB security group does not allow deletion.

HTTP Status Code: 400

InvalidVPCNetworkStateFault

DB subnet group does not cover all Availability Zones after it is created because users' change.

HTTP Status Code: 400

OptionGroupNotFoundFault

HTTP Status Code: 404

ProvisionedIopsNotAvailableInAZFault

Provisioned IOPS not available in the specified Availability Zone.

HTTP Status Code: 400

StorageQuotaExceeded

Request would result in user exceeding the allowed amount of storage available across all DB instances.

HTTP Status Code: 400
**StorageTypeNotSupported**

*StorageType* specified cannot be associated with the DB Instance.

HTTP Status Code: 400

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
ModifyDBParameterGroup

Modifies the parameters of a DB parameter group. To modify more than one parameter, submit a list of the following: ParameterName, ParameterValue, and ApplyMethod. A maximum of 20 parameters can be modified in a single request.

**Note**
Changes to dynamic parameters are applied immediately. Changes to static parameters require a reboot without failover to the DB instance associated with the parameter group before the change can take effect.

**Important**
After you modify a DB parameter group, you should wait at least 5 minutes before creating your first DB instance that uses that DB parameter group as the default parameter group. This allows Amazon Neptune to fully complete the modify action before the parameter group is used as the default for a new DB instance. This is especially important for parameters that are critical when creating the default database for a DB instance, such as the character set for the default database defined by the character_set_database parameter. You can use the Parameter Groups option of the Amazon Neptune console or the DescribeDBParameters command to verify that your DB parameter group has been created or modified.

**Request Parameters**

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

**DBParameterGroupName**

The name of the DB parameter group.

Constraints:
- If supplied, must match the name of an existing DBParameterGroup.

Type: String

Required: Yes

**Parameters.Parameter.N**

An array of parameter names, values, and the apply method for the parameter update. At least one parameter name, value, and apply method must be supplied; subsequent arguments are optional. A maximum of 20 parameters can be modified in a single request.

Valid Values (for the application method): immediate | pending-reboot

**Note**
You can use the immediate value with dynamic parameters only. You can use the pending-reboot value for both dynamic and static parameters, and changes are applied when you reboot the DB instance without failover.

Type: Array of Parameter (p. 347) objects

Required: Yes

**Response Elements**

The following element is returned by the service.
DBParameterGroupName

Provides the name of the DB parameter group.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

DBParameterGroupNameNotFound

DBParameterGroupName does not refer to an existing DB parameter group.

HTTP Status Code: 404

InvalidDBParameterGroupState

The DB parameter group is in use or is in an invalid state. If you are attempting to delete the parameter group, you cannot delete it when the parameter group is in this state.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
ModifyDBSubnetGroup

Modifies an existing DB subnet group. DB subnet groups must contain at least one subnet in at least two AZs in the AWS Region.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

**DBSubnetGroupDescription**

The description for the DB subnet group.

Type: String

Required: No

**DBSubnetGroupName**

The name for the DB subnet group. This value is stored as a lowercase string. You can't modify the default subnet group.

Constraints: Must match the name of an existing DBSubnetGroup. Must not be default.

Example: mySubnetgroup

Type: String

Required: Yes

**SubnetIds.SubnetIdentifier.N**

The EC2 subnet IDs for the DB subnet group.

Type: Array of strings

Required: Yes

Response Elements

The following element is returned by the service.

**DBSubnetGroup**

Contains the details of an Amazon Neptune DB subnet group.

This data type is used as a response element in the DescribeDBSubnetGroups (p. 226) action.

Type: DBSubnetGroup (p. 331) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

**DBSubnetGroupDoesNotCoverEnoughAZs**

Subnets in the DB subnet group should cover at least two Availability Zones unless there is only one Availability Zone.
HTTP Status Code: 400

**DBSubnetGroupNameNotFoundFault**

*DBSubnetGroupName* does not refer to an existing DB subnet group.

HTTP Status Code: 404

**DBSubnetQuotaExceededFault**

Request would result in user exceeding the allowed number of subnets in a DB subnet groups.

HTTP Status Code: 400

**InvalidSubnet**

The requested subnet is invalid, or multiple subnets were requested that are not all in a common VPC.

HTTP Status Code: 400

**SubnetAlreadyInUse**

The DB subnet is already in use in the Availability Zone.

HTTP Status Code: 400

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
ModifyEventSubscription

Modifies an existing event notification subscription. Note that you can't modify the source identifiers using this call; to change source identifiers for a subscription, use the AddSourceIdentifierToSubscription (p. 150) and RemoveSourceIdentifierFromSubscription (p. 282) calls.

You can see a list of the event categories for a given SourceType by using the DescribeEventCategories action.

**Request Parameters**

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

**Enabled**

A Boolean value; set to true to activate the subscription.

Type: Boolean

Required: No

**EventCategories.EventCategory.N**

A list of event categories for a SourceType that you want to subscribe to. You can see a list of the categories for a given SourceType by using the DescribeEventCategories action.

Type: Array of strings

Required: No

**SnsTopicArn**

The Amazon Resource Name (ARN) of the SNS topic created for event notification. The ARN is created by Amazon SNS when you create a topic and subscribe to it.

Type: String

Required: No

**SourceType**

The type of source that is generating the events. For example, if you want to be notified of events generated by a DB instance, you would set this parameter to db-instance. If this value is not specified, all events are returned.

Valid values: db-instance | db-parameter-group | db-security-group | db-snapshot

Type: String

Required: No

**SubscriptionName**

The name of the event notification subscription.

Type: String

Required: Yes

**Response Elements**

The following element is returned by the service.
EventSubscription

Contains the results of a successful invocation of the DescribeEventSubscriptions (p. 237) action.

Type: EventSubscription (p. 340) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

EventSubscriptionQuotaExceeded

HTTP Status Code: 400

SNSInvalidTopic

HTTP Status Code: 400

SNSNoAuthorization

HTTP Status Code: 400

SNSTopicArnNotFound

HTTP Status Code: 404

SubscriptionCategoryNotFound

HTTP Status Code: 404

SubscriptionNotFound

HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
PromoteReadReplicaDBCluster

Promotes a Read Replica DB cluster to a standalone DB cluster.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

DBClusterIdentifier

The identifier of the DB cluster Read Replica to promote. This parameter is not case-sensitive.

Constraints:

• Must match the identifier of an existing DBCluster Read Replica.

Example: my-cluster-replica1

Type: String

Required: Yes

Response Elements

The following element is returned by the service.

DBCluster

Contains the details of an Amazon Neptune DB cluster.

This data type is used as a response element in the DescribeDBClusters (p. 210) action.

Type: DBCluster (p. 304) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

DBClusterNotFoundFault

$DBClusterIdentifier$ does not refer to an existing DB cluster.

HTTP Status Code: 404

InvalidDBClusterStateFault

The DB cluster is not in a valid state.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

• AWS Command Line Interface
• AWS SDK for .NET
• AWS SDK for C++
• AWS SDK for Go
• AWS SDK for Java
• AWS SDK for JavaScript
• AWS SDK for PHP V3
• AWS SDK for Python
• AWS SDK for Ruby V2
RebootDBInstance

You might need to reboot your DB instance, usually for maintenance reasons. For example, if you make certain modifications, or if you change the DB parameter group associated with the DB instance, you must reboot the instance for the changes to take effect.

Rebooting a DB instance restarts the database engine service. Rebooting a DB instance results in a momentary outage, during which the DB instance status is set to rebooting.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

**DBInstanceIdentifier**

The DB instance identifier. This parameter is stored as a lowercase string.

Constraints:
- Must match the identifier of an existing DBInstance.

Type: String

Required: Yes

**ForceFailover**

When true, the reboot is conducted through a MultiAZ failover.

Constraint: You can't specify true if the instance is not configured for MultiAZ.

Type: Boolean

Required: No

Response Elements

The following element is returned by the service.

**DBInstance**

Contains the details of an Amazon Neptune DB instance.

This data type is used as a response element in the DescribeDBInstances (p. 220) action.

Type: DBInstance (p. 320) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

**DBInstanceNotFound**

DBInstanceIdentifier does not refer to an existing DB instance.

HTTP Status Code: 404
InvalidDBInstanceState

The specified DB instance is not in the available state.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
RemoveRoleFromDBCluster

Disassociates an Identity and Access Management (IAM) role from a DB cluster.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

**DBClusterIdentifier**

The name of the DB cluster to disassociate the IAM role from.

Type: String

Required: Yes

**RoleArn**

The Amazon Resource Name (ARN) of the IAM role to disassociate from the DB cluster, for example `arn:aws:iam::123456789012:role/NeptuneAccessRole`.

Type: String

Required: Yes

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

**DBClusterNotFoundFault**

*DBClusterIdentifier* does not refer to an existing DB cluster.

HTTP Status Code: 404

**DBClusterRoleNotFound**

The specified IAM role Amazon Resource Name (ARN) is not associated with the specified DB cluster.

HTTP Status Code: 404

**InvalidDBClusterStateFault**

The DB cluster is not in a valid state.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
• AWS SDK for PHP V3
• AWS SDK for Python
• AWS SDK for Ruby V2
RemoveSourceIdentifierFromSubscription

Removes a source identifier from an existing event notification subscription.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

SourceIdentifier

The source identifier to be removed from the subscription, such as the DB instance identifier for a DB instance or the name of a security group.

Type: String
Required: Yes

SubscriptionName

The name of the event notification subscription you want to remove a source identifier from.

Type: String
Required: Yes

Response Elements

The following element is returned by the service.

EventSubscription

Contains the results of a successful invocation of the DescribeEventSubscriptions (p. 237) action.

Type: EventSubscription (p. 340) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

SourceNotFound

HTTP Status Code: 404

SubscriptionNotFound

HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
• AWS SDK for Java
• AWS SDK for JavaScript
• AWS SDK for PHP V3
• AWS SDK for Python
• AWS SDK for Ruby V2
RemoveTagsFromResource

Removes metadata tags from an Amazon Neptune resource.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

ResourceName

The Amazon Neptune resource that the tags are removed from. This value is an Amazon Resource Name (ARN). For information about creating an ARN, see Constructing an Amazon Resource Name (ARN).

Type: String
Required: Yes

TagKeys.member.N

The tag key (name) of the tag to be removed.

Type: Array of strings
Required: Yes

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

DBClusterNotFoundFault

DBClusterIdentifier does not refer to an existing DB cluster.

HTTP Status Code: 404

DBInstanceNotFound

DBInstanceIdentifier does not refer to an existing DB instance.

HTTP Status Code: 404

DBSnapshotNotFound

DBSnapshotIdentifier does not refer to an existing DB snapshot.

HTTP Status Code: 404

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
RemoveTagsFromResource

- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
ResetDBClusterParameterGroup

Modifies the parameters of a DB cluster parameter group to the default value. To reset specific parameters submit a list of the following: ParameterName and ApplyMethod. To reset the entire DB cluster parameter group, specify the DBClusterParameterGroupName and ResetAllParameters parameters.

When resetting the entire group, dynamic parameters are updated immediately and static parameters are set to pending-reboot to take effect on the next DB instance restart or RebootDBInstance (p. 278) request. You must call RebootDBInstance (p. 278) for every DB instance in your DB cluster that you want the updated static parameter to apply to.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

DBClusterParameterGroupName

The name of the DB cluster parameter group to reset.

Type: String

Required: Yes

Parameters.Parameter.N

A list of parameter names in the DB cluster parameter group to reset to the default values. You can't use this parameter if the ResetAllParameters parameter is set to true.

Type: Array of Parameter (p. 347) objects

Required: No

ResetAllParameters

A value that is set to true to reset all parameters in the DB cluster parameter group to their default values, and false otherwise. You can't use this parameter if there is a list of parameter names specified for the Parameters parameter.

Type: Boolean

Required: No

Response Elements

The following element is returned by the service.

DBClusterParameterGroupName

The name of the DB cluster parameter group.

Constraints:
- Must be 1 to 255 letters or numbers.
- First character must be a letter
- Cannot end with a hyphen or contain two consecutive hyphens

Note
This value is stored as a lowercase string.
Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

**DBParameterGroupNotFound**

*DBParameterGroupName* does not refer to an existing DB parameter group.

HTTP Status Code: 404

**InvalidDBParameterGroupState**

The DB parameter group is in use or is in an invalid state. If you are attempting to delete the parameter group, you cannot delete it when the parameter group is in this state.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
ResetDBParameterGroup

Modifies the parameters of a DB parameter group to the engine/system default value. To reset specific parameters, provide a list of the following: ParameterName and ApplyMethod. To reset the entire DB parameter group, specify the DBParameterGroup name and ResetAllParameters parameters. When resetting the entire group, dynamic parameters are updated immediately and static parameters are set to pending-reboot to take effect on the next DB instance restart or RebootDBInstance request.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

**DBParameterGroupName**

The name of the DB parameter group.

Constraints:

- Must match the name of an existing DBParameterGroup.

Type: String

Required: Yes

**Parameters.Parameter.N**

To reset the entire DB parameter group, specify the DBParameterGroup name and ResetAllParameters parameters. To reset specific parameters, provide a list of the following: ParameterName and ApplyMethod. A maximum of 20 parameters can be modified in a single request.

Valid Values (for Apply method): pending-reboot

Type: Array of Parameter (p. 347) objects

Required: No

**ResetAllParameters**

Specifies whether (true) or not (false) to reset all parameters in the DB parameter group to default values.

Default: true

Type: Boolean

Required: No

Response Elements

The following element is returned by the service.

**DBParameterGroupName**

Provides the name of the DB parameter group.

Type: String
Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

DBParameterGroupNotFound

`DBParameterGroupName` does not refer to an existing DB parameter group.

HTTP Status Code: 404

InvalidDBParameterGroupState

The DB parameter group is in use or is in an invalid state. If you are attempting to delete the parameter group, you cannot delete it when the parameter group is in this state.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
RestoreDBClusterFromSnapshot

Creates a new DB cluster from a DB snapshot or DB cluster snapshot.

If a DB snapshot is specified, the target DB cluster is created from the source DB snapshot with a default configuration and default security group.

If a DB cluster snapshot is specified, the target DB cluster is created from the source DB cluster restore point with the same configuration as the original source DB cluster, except that the new DB cluster is created with the default security group.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

AvailabilityZones.AvailabilityZone.N

Provides the list of EC2 Availability Zones that instances in the restored DB cluster can be created in.

Type: Array of strings

Required: No

DatabaseName

The database name for the restored DB cluster.

Type: String

Required: No

DBClusterIdentifier

The name of the DB cluster to create from the DB snapshot or DB cluster snapshot. This parameter isn't case-sensitive.

Constraints:
- Must contain from 1 to 63 letters, numbers, or hyphens
- First character must be a letter
- Cannot end with a hyphen or contain two consecutive hyphens

Example: my-snapshot-id

Type: String

Required: Yes

DBSubnetGroupName

The name of the DB subnet group to use for the new DB cluster.

Constraints: If supplied, must match the name of an existing DBSubnetGroup.

Example: mySubnetgroup

Type: String

Required: No
EnableIAMDatabaseAuthentication

True to enable mapping of AWS Identity and Access Management (IAM) accounts to database accounts, and otherwise false.

Default: false
Type: Boolean
Required: No

Engine

The database engine to use for the new DB cluster.

Default: The same as source
Constraint: Must be compatible with the engine of the source
Type: String
Required: Yes

EngineVersion

The version of the database engine to use for the new DB cluster.

Type: String
Required: No

KmsKeyId

The AWS KMS key identifier to use when restoring an encrypted DB cluster from a DB snapshot or DB cluster snapshot.

The KMS key identifier is the Amazon Resource Name (ARN) for the KMS encryption key. If you are restoring a DB cluster with the same AWS account that owns the KMS encryption key used to encrypt the new DB cluster, then you can use the KMS key alias instead of the ARN for the KMS encryption key.

If you do not specify a value for the KmsKeyId parameter, then the following will occur:

- If the DB snapshot or DB cluster snapshot in SnapshotIdentifier is encrypted, then the restored DB cluster is encrypted using the KMS key that was used to encrypt the DB snapshot or DB cluster snapshot.
- If the DB snapshot or DB cluster snapshot in SnapshotIdentifier is not encrypted, then the restored DB cluster is not encrypted.

Type: String
Required: No

OptionGroupName

The name of the option group to use for the restored DB cluster.

Type: String
Required: No

Port

The port number on which the new DB cluster accepts connections.

Constraints: Value must be 1150–65535
Default: The same port as the original DB cluster.

Type: Integer

Required: No

**SnapshotIdentifier**

The identifier for the DB snapshot or DB cluster snapshot to restore from.

You can use either the name or the Amazon Resource Name (ARN) to specify a DB cluster snapshot. However, you can use only the ARN to specify a DB snapshot.

Constraints:
- Must match the identifier of an existing Snapshot.

Type: String

Required: Yes

**Tags.Tag.N**

The tags to be assigned to the restored DB cluster.

Type: Array of Tag (p. 358) objects

Required: No

**VpcSecurityGroupIds.VpcSecurityGroupId.N**

A list of VPC security groups that the new DB cluster will belong to.

Type: Array of strings

Required: No

**Response Elements**

The following element is returned by the service.

**DBCluster**

Contains the details of an Amazon Neptune DB cluster.

This data type is used as a response element in the DescribeDBClusters (p. 210) action.

Type: DBCluster (p. 304) object

**Errors**

For information about the errors that are common to all actions, see Common Errors (p. 363).

**DBClusterAlreadyExistsFault**

User already has a DB cluster with the given identifier.

HTTP Status Code: 400

**DBClusterQuotaExceededFault**

User attempted to create a new DB cluster and the user has already reached the maximum allowed DB cluster quota.
HTTP Status Code: 403

**DBClusterSnapshotNotFoundFault**

*DBClusterSnapshotIdentifier* does not refer to an existing DB cluster snapshot.

HTTP Status Code: 404

**DBSnapshotNotFound**

*DBSnapshotIdentifier* does not refer to an existing DB snapshot.

HTTP Status Code: 404

**DBSubnetGroupNameNotFoundFault**

*DBSubnetGroupName* does not refer to an existing DB subnet group.

HTTP Status Code: 404

**DBSubnetGroupNameNotFoundFault**

*DBSubnetGroupName* does not refer to an existing DB subnet group.

HTTP Status Code: 404

**InsufficientDBClusterCapacityFault**

The DB cluster does not have enough capacity for the current operation.

HTTP Status Code: 403

**InsufficientStorageClusterCapacity**

There is insufficient storage available for the current action. You may be able to resolve this error by updating your subnet group to use different Availability Zones that have more storage available.

HTTP Status Code: 400

**InvalidDBClusterSnapshotStateFault**

The supplied value is not a valid DB cluster snapshot state.

HTTP Status Code: 400

**InvalidDBSnapshotState**

The state of the DB snapshot does not allow deletion.

HTTP Status Code: 400

**InvalidRestoreFault**

Cannot restore from vpc backup to non-vpc DB instance.

HTTP Status Code: 400

**InvalidSubnet**

The requested subnet is invalid, or multiple subnets were requested that are not all in a common VPC.

HTTP Status Code: 400

**InvalidVPCNetworkStateFault**

DB subnet group does not cover all Availability Zones after it is created because users' change.

HTTP Status Code: 400
KMSKeyNotAccessibleFault

Error accessing KMS key.

HTTP Status Code: 400

OptionGroupNotFoundFault

HTTP Status Code: 404

StorageQuotaExceeded

Request would result in user exceeding the allowed amount of storage available across all DB instances.

HTTP Status Code: 400

StorageQuotaExceeded

Request would result in user exceeding the allowed amount of storage available across all DB instances.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
**RestoreDBClusterToPointInTime**

Restores a DB cluster to an arbitrary point in time. Users can restore to any point in time before `LatestRestorableTime` for up to `BackupRetentionPeriod` days. The target DB cluster is created from the source DB cluster with the same configuration as the original DB cluster, except that the new DB cluster is created with the default DB security group.

**Note**

This action only restores the DB cluster, not the DB instances for that DB cluster. You must invoke the `CreateDBInstance` action to create DB instances for the restored DB cluster, specifying the identifier of the restored DB cluster in `DBClusterIdentifier`. You can create DB instances only after the `RestoreDBClusterToPointInTime` action has completed and the DB cluster is available.

**Request Parameters**

For information about the parameters that are common to all actions, see Common Parameters (p. 365).

**DBClusterIdentifier**

The name of the new DB cluster to be created.

Constraints:
- Must contain from 1 to 63 letters, numbers, or hyphens
- First character must be a letter
- Cannot end with a hyphen or contain two consecutive hyphens

Type: String

Required: Yes

**DBSubnetGroupName**

The DB subnet group name to use for the new DB cluster.

Constraints: If supplied, must match the name of an existing `DBSubnetGroup`.

Example: `mySubnetgroup`

Type: String

Required: No

**EnableIAMDatabaseAuthentication**

True to enable mapping of AWS Identity and Access Management (IAM) accounts to database accounts, and otherwise false.

Default: `false`

Type: Boolean

Required: No

**KmsKeyId**

The AWS KMS key identifier to use when restoring an encrypted DB cluster from an encrypted DB cluster.

The KMS key identifier is the Amazon Resource Name (ARN) for the KMS encryption key. If you are restoring a DB cluster with the same AWS account that owns the KMS encryption key used to
encrypt the new DB cluster, then you can use the KMS key alias instead of the ARN for the KMS encryption key.

You can restore to a new DB cluster and encrypt the new DB cluster with a KMS key that is different than the KMS key used to encrypt the source DB cluster. The new DB cluster is encrypted with the KMS key identified by the KmsKeyId parameter.

If you do not specify a value for the KmsKeyId parameter, then the following will occur:
• If the DB cluster is encrypted, then the restored DB cluster is encrypted using the KMS key that was used to encrypt the source DB cluster.
• If the DB cluster is not encrypted, then the restored DB cluster is not encrypted.

If DBClusterIdentifier refers to a DB cluster that is not encrypted, then the restore request is rejected.

Type: String
Required: No

OptionGroupName

The name of the option group for the new DB cluster.

Type: String
Required: No

Port

The port number on which the new DB cluster accepts connections.

Constraints: Value must be 1150–65535

Default: The same port as the original DB cluster.

Type: Integer
Required: No

RestoreToTime

The date and time to restore the DB cluster to.

Valid Values: Value must be a time in Universal Coordinated Time (UTC) format

Constraints:
• Must be before the latest restorable time for the DB instance
• Must be specified if UseLatestRestorableTime parameter is not provided
• Cannot be specified if UseLatestRestorableTime parameter is true
• Cannot be specified if RestoreType parameter is copy-on-write

Example: 2015-03-07T23:45:00Z

Type: Timestamp
Required: No

RestoreType

The type of restore to be performed. You can specify one of the following values:
• full-copy - The new DB cluster is restored as a full copy of the source DB cluster.
• copy-on-write - The new DB cluster is restored as a clone of the source DB cluster.

Constraints: You can't specify copy-on-write if the engine version of the source DB cluster is earlier than 1.11.

If you don't specify a RestoreType value, then the new DB cluster is restored as a full copy of the source DB cluster.

Type: String
Required: No

**SourceDBClusterIdentifier**

The identifier of the source DB cluster from which to restore.

Constraints:
- Must match the identifier of an existing DBCluster.

Type: String
Required: Yes

**Tags.Tag.N**

A list of tags. For more information, see Tagging Amazon Neptune Resources.

Type: Array of Tag (p. 358) objects
Required: No

**UseLatestRestorableTime**

A value that is set to true to restore the DB cluster to the latest restorable backup time, and false otherwise.

Default: false

Constraints: Cannot be specified if RestoreToTime parameter is provided.

Type: Boolean
Required: No

**VpcSecurityGroupIds.VpcSecurityGroupId.N**

A list of VPC security groups that the new DB cluster belongs to.

Type: Array of strings
Required: No

## Response Elements

The following element is returned by the service.

**DBCluster**

Contains the details of an Amazon Neptune DB cluster.

This data type is used as a response element in the DescribeDBClusters (p. 210) action.

Type: DBCluster (p. 304) object
Errors

For information about the errors that are common to all actions, see Common Errors (p. 363).

**DBClusterAlreadyExistsFault**

User already has a DB cluster with the given identifier.

HTTP Status Code: 400

**DBClusterNotFoundFault**

*DBClusterIdentifier* does not refer to an existing DB cluster.

HTTP Status Code: 404

**DBClusterQuotaExceededFault**

User attempted to create a new DB cluster and the user has already reached the maximum allowed DB cluster quota.

HTTP Status Code: 403

**DBClusterSnapshotNotFoundFault**

*DBClusterSnapshotIdentifier* does not refer to an existing DB cluster snapshot.

HTTP Status Code: 404

**DBSubnetGroupNotFoundFault**

*DBSubnetGroupName* does not refer to an existing DB subnet group.

HTTP Status Code: 404

**InsufficientDBClusterCapacityFault**

The DB cluster does not have enough capacity for the current operation.

HTTP Status Code: 403

**InsufficientStorageClusterCapacity**

There is insufficient storage available for the current action. You may be able to resolve this error by updating your subnet group to use different Availability Zones that have more storage available.

HTTP Status Code: 400

**InvalidDBClusterSnapshotStateFault**

The supplied value is not a valid DB cluster snapshot state.

HTTP Status Code: 400

**InvalidDBClusterStateFault**

The DB cluster is not in a valid state.

HTTP Status Code: 400

**InvalidDBSnapshotState**

The state of the DB snapshot does not allow deletion.

HTTP Status Code: 400

**InvalidRestoreFault**

Cannot restore from vpc backup to non-vpc DB instance.
HTTP Status Code: 400

InvalidSubnet

The requested subnet is invalid, or multiple subnets were requested that are not all in a common VPC.

HTTP Status Code: 400

InvalidVPCNetworkStateFault

DB subnet group does not cover all Availability Zones after it is created because users' change.

HTTP Status Code: 400

KMSKeyNotAccessibleFault

Error accessing KMS key.

HTTP Status Code: 400

OptionGroupNotFoundFault

HTTP Status Code: 404

StorageQuotaExceeded

Request would result in user exceeding the allowed amount of storage available across all DB instances.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

Data Types

The following data types are supported:

- AvailabilityZone (p. 301)
- CharacterSet (p. 302)
- CloudwatchLogsExportConfiguration (p. 303)
- DBCluster (p. 304)
- DBClusterMember (p. 309)
- DBClusterOptionGroupStatus (p. 310)
- DBClusterParameterGroup (p. 311)
- DBClusterRole (p. 312)
- DBClusterSnapshot (p. 313)
- DBClusterSnapshotAttribute (p. 316)
- DBClusterSnapshotAttributesResult (p. 317)
- DBEngineVersion (p. 318)
- DBInstance (p. 320)
- DBInstanceStatusInfo (p. 327)
- DBParameterGroup (p. 328)
- DBParameterGroupStatus (p. 329)
- DBSecurityGroupMembership (p. 330)
- DBSubnetGroup (p. 331)
- DomainMembership (p. 333)
- DoubleRange (p. 334)
- Endpoint (p. 335)
- EngineDefaults (p. 336)
- Event (p. 337)
- EventCategoriesMap (p. 339)
- EventSubscription (p. 340)
- Filter (p. 342)
- OptionGroupMembership (p. 343)
- OrderableDBInstanceOption (p. 344)
- Parameter (p. 347)
- PendingCloudwatchLogsExports (p. 349)
- PendingMaintenanceAction (p. 350)
- PendingModifiedValues (p. 352)
- Range (p. 355)
- ResourcePendingMaintenanceActions (p. 356)
- Subnet (p. 357)
- Tag (p. 358)
- Timezone (p. 359)
- UpgradeTarget (p. 360)
- ValidDBInstanceModificationsMessage (p. 361)
- ValidStorageOptions (p. 362)
- VpcSecurityGroupMembership (p. 363)
AvailabilityZone

Contains Availability Zone information.

This data type is used as an element in the following data type:

- OrderableDBInstanceOption (p. 344)

Contents

Name

The name of the availability zone.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
CharacterSet

This data type is used as a response element in the action DescribeDBEngineVersions (p. 217).

Contents

**CharacterSetDescription**

The description of the character set.

Type: String

Required: No

**CharacterSetName**

The name of the character set.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
CloudwatchLogsExportConfiguration

The configuration setting for the log types to be enabled for export to CloudWatch Logs for a specific DB instance or DB cluster.

Contents

DisableLogTypes.member.N

The list of log types to disable.

Type: Array of strings

Required: No

EnableLogTypes.member.N

The list of log types to enable.

Type: Array of strings

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
DBCluster

Contains the details of an Amazon Neptune DB cluster.

This data type is used as a response element in the DescribeDBClusters (p. 210) action.

Contents

AllocatedStorage

AllocatedStorage always returns 1, because Neptune DB cluster storage size is not fixed, but instead automatically adjusts as needed.

Type: Integer

Required: No

AssociatedRoles.DBClusterRole.N

Provides a list of the AWS Identity and Access Management (IAM) roles that are associated with the DB cluster. IAM roles that are associated with a DB cluster grant permission for the DB cluster to access other AWS services on your behalf.

Type: Array of DBClusterRole (p. 312) objects

Required: No

AvailabilityZones.AvailabilityZone.N

Provides the list of EC2 Availability Zones that instances in the DB cluster can be created in.

Type: Array of strings

Required: No

BackupRetentionPeriod

Specifies the number of days for which automatic DB snapshots are retained.

Type: Integer

Required: No

CharacterSetName

If present, specifies the name of the character set that this cluster is associated with.

Type: String

Required: No

CloneGroupId

Identifies the clone group to which the DB cluster is associated.

Type: String

Required: No

ClusterCreateTime

Specifies the time when the DB cluster was created, in Universal Coordinated Time (UTC).

Type: Timestamp
Required: No

**DatabaseName**

Contains the name of the initial database of this DB cluster that was provided at create time, if one was specified when the DB cluster was created. This same name is returned for the life of the DB cluster.

Type: String

Required: No

**DBClusterArn**

The Amazon Resource Name (ARN) for the DB cluster.

Type: String

Required: No

**DBClusterIdentifier**

Contains a user-supplied DB cluster identifier. This identifier is the unique key that identifies a DB cluster.

Type: String

Required: No

**DBClusterMembers.DBClusterMember.N**

Provides the list of instances that make up the DB cluster.

Type: Array of DBClusterMember (p. 309) objects

Required: No

**DBClusterOptionGroupMemberships.DBClusterOptionGroup.N**

Provides the list of option group memberships for this DB cluster.

Type: Array of DBClusterOptionGroupStatus (p. 310) objects

Required: No

**DBClusterParameterGroup**

Specifies the name of the DB cluster parameter group for the DB cluster.

Type: String

Required: No

**DbClusterResourceId**

The AWS Region-unique, immutable identifier for the DB cluster. This identifier is found in AWS CloudTrail log entries whenever the AWS KMS key for the DB cluster is accessed.

Type: String

Required: No

**DBSubnetGroup**

Specifies information on the subnet group associated with the DB cluster, including the name, description, and subnets in the subnet group.
Type: String
Required: No

**EarliestRestorableTime**

Specifies the earliest time to which a database can be restored with point-in-time restore.

Type: Timestamp
Required: No

**Endpoint**

Specifies the connection endpoint for the primary instance of the DB cluster.

Type: String
Required: No

**Engine**

Provides the name of the database engine to be used for this DB cluster.

Type: String
Required: No

**EngineVersion**

Indicates the database engine version.

Type: String
Required: No

**HostedZoneId**

Specifies the ID that Amazon Route 53 assigns when you create a hosted zone.

Type: String
Required: No

**IAMDatabaseAuthenticationEnabled**

True if mapping of AWS Identity and Access Management (IAM) accounts to database accounts is enabled, and otherwise false.

Type: Boolean
Required: No

**KmsKeyId**

If `StorageEncrypted` is true, the AWS KMS key identifier for the encrypted DB cluster.

Type: String
Required: No

**LatestRestorableTime**

Specifies the latest time to which a database can be restored with point-in-time restore.

Type: Timestamp
Required: No
**MasterUsername**

Contains the master username for the DB cluster.

Type: String

Required: No

**MultiAZ**

Specifies whether the DB cluster has instances in multiple Availability Zones.

Type: Boolean

Required: No

**PercentProgress**

Specifies the progress of the operation as a percentage.

Type: String

Required: No

**Port**

Specifies the port that the database engine is listening on.

Type: Integer

Required: No

**PreferredBackupWindow**

Specifies the daily time range during which automated backups are created if automated backups are enabled, as determined by the `BackupRetentionPeriod`.

Type: String

Required: No

**PreferredMaintenanceWindow**

Specifies the weekly time range during which system maintenance can occur, in Universal Coordinated Time (UTC).

Type: String

Required: No

**ReaderEndpoint**

The reader endpoint for the DB cluster. The reader endpoint for a DB cluster load-balances connections across the Read Replicas that are available in a DB cluster. As clients request new connections to the reader endpoint, Neptune distributes the connection requests among the Read Replicas in the DB cluster. This functionality can help balance your read workload across multiple Read Replicas in your DB cluster.

If a failover occurs, and the Read Replica that you are connected to is promoted to be the primary instance, your connection is dropped. To continue sending your read workload to other Read Replicas in the cluster, you can then reconnect to the reader endpoint.

Type: String

Required: No
**ReadReplicaIdentifiers.ReadReplicaIdentifier.N**

Contains one or more identifiers of the Read Replicas associated with this DB cluster.

Type: Array of strings

Required: No

**ReplicationSourceIdentifier**

Contains the identifier of the source DB cluster if this DB cluster is a Read Replica.

Type: String

Required: No

**Status**

Specifies the current state of this DB cluster.

Type: String

Required: No

**StorageEncrypted**

Specifies whether the DB cluster is encrypted.

Type: Boolean

Required: No


Provides a list of VPC security groups that the DB cluster belongs to.

Type: Array of VpcSecurityGroupMembership (p. 363) objects

Required: No

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
DBClusterMember

Contains information about an instance that is part of a DB cluster.

Contents

DBClusterParameterGroupStatus

Specifies the status of the DB cluster parameter group for this member of the DB cluster.

Type: String
Required: No

DBInstanceIdentifier

Specifies the instance identifier for this member of the DB cluster.

Type: String
Required: No

IsClusterWriter

Value that is true if the cluster member is the primary instance for the DB cluster and false otherwise.

Type: Boolean
Required: No

PromotionTier

A value that specifies the order in which a Read Replica is promoted to the primary instance after a failure of the existing primary instance.

Type: Integer
Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
DBClusterOptionGroupStatus

Contains status information for a DB cluster option group.

Contents

DBClusterOptionGroupName

Specifies the name of the DB cluster option group.

Type: String

Required: No

Status

Specifies the status of the DB cluster option group.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
DBClusterParameterGroup

Contains the details of an Amazon Neptune DB cluster parameter group.
This data type is used as a response element in the DescribeDBClusterParameterGroups (p. 206) action.

Contents

DBClusterParameterGroupArn

The Amazon Resource Name (ARN) for the DB cluster parameter group.
Type: String
Required: No

DBClusterParameterGroupName

Provides the name of the DB cluster parameter group.
Type: String
Required: No

DBParameterGroupFamily

Provides the name of the DB parameter group family that this DB cluster parameter group is compatible with.
Type: String
Required: No

Description

Provides the customer-specified description for this DB cluster parameter group.
Type: String
Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
DBClusterRole

Describes an AWS Identity and Access Management (IAM) role that is associated with a DB cluster.

Contents

RoleArn

The Amazon Resource Name (ARN) of the IAM role that is associated with the DB cluster.

Type: String
Required: No

Status

Describes the state of association between the IAM role and the DB cluster. The Status property returns one of the following values:

- **ACTIVE** - the IAM role ARN is associated with the DB cluster and can be used to access other AWS services on your behalf.
- **PENDING** - the IAM role ARN is being associated with the DB cluster.
- **INVALID** - the IAM role ARN is associated with the DB cluster, but the DB cluster is unable to assume the IAM role in order to access other AWS services on your behalf.

Type: String
Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
DBClusterSnapshot

Contains the details for an Amazon Neptune DB cluster snapshot

This data type is used as a response element in the DescribeDBClusterSnapshots (p. 214) action.

Contents

AllocatedStorage

Specifies the allocated storage size in gibibytes (GiB).

Type: Integer

Required: No

AvailabilityZones.AvailabilityZone.N

Provides the list of EC2 Availability Zones that instances in the DB cluster snapshot can be restored in.

Type: Array of strings

Required: No

ClusterCreateTime

Specifies the time when the DB cluster was created, in Universal Coordinated Time (UTC).

Type: Timestamp

Required: No

DBClusterIdentifier

Specifies the DB cluster identifier of the DB cluster that this DB cluster snapshot was created from.

Type: String

Required: No

DBClusterSnapshotArn

The Amazon Resource Name (ARN) for the DB cluster snapshot.

Type: String

Required: No

DBClusterSnapshotIdentifier

Specifies the identifier for the DB cluster snapshot.

Type: String

Required: No

Engine

Specifies the name of the database engine.

Type: String

Required: No
**EngineVersion**

Provides the version of the database engine for this DB cluster snapshot.

Type: String

Required: No

**IAMDatabaseAuthenticationEnabled**

True if mapping of AWS Identity and Access Management (IAM) accounts to database accounts is enabled, and otherwise false.

Type: Boolean

Required: No

**KmsKeyId**

If `StorageEncrypted` is true, the AWS KMS key identifier for the encrypted DB cluster snapshot.

Type: String

Required: No

**LicenseModel**

Provides the license model information for this DB cluster snapshot.

Type: String

Required: No

**MasterUsername**

Provides the master username for the DB cluster snapshot.

Type: String

Required: No

**PercentProgress**

Specifies the percentage of the estimated data that has been transferred.

Type: Integer

Required: No

**Port**

Specifies the port that the DB cluster was listening on at the time of the snapshot.

Type: Integer

Required: No

**SnapshotCreateTime**

Provides the time when the snapshot was taken, in Universal Coordinated Time (UTC).

Type: Timestamp

Required: No

**SnapshotType**

Provides the type of the DB cluster snapshot.
Type: String
Required: No

**SourceDBClusterSnapshotArn**
If the DB cluster snapshot was copied from a source DB cluster snapshot, the Amazon Resource Name (ARN) for the source DB cluster snapshot, otherwise, a null value.
Type: String
Required: No

**Status**
Specifies the status of this DB cluster snapshot.
Type: String
Required: No

**StorageEncrypted**
Specifies whether the DB cluster snapshot is encrypted.
Type: Boolean
Required: No

**VpcId**
Provides the VPC ID associated with the DB cluster snapshot.
Type: String
Required: No

**See Also**
For more information about using this API in one of the language-specific AWS SDKs, see the following:
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
DBClusterSnapshotAttribute

Contains the name and values of a manual DB cluster snapshot attribute.

Manual DB cluster snapshot attributes are used to authorize other AWS accounts to restore a manual DB cluster snapshot. For more information, see the ModifyDBClusterSnapshotAttribute (p. 257) API action.

Contents

AttributeName

The name of the manual DB cluster snapshot attribute.

The attribute named restore refers to the list of AWS accounts that have permission to copy or restore the manual DB cluster snapshot. For more information, see the ModifyDBClusterSnapshotAttribute (p. 257) API action.

Type: String

Required: No

AttributeValue.N

The value(s) for the manual DB cluster snapshot attribute.

If the AttributeName field is set to restore, then this element returns a list of IDs of the AWS accounts that are authorized to copy or restore the manual DB cluster snapshot. If a value of all is in the list, then the manual DB cluster snapshot is public and available for any AWS account to copy or restore.

Type: Array of strings

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
DBClusterSnapshotAttributesResult

Contains the results of a successful call to the DescribeDBClusterSnapshotAttributes (p. 212) API action.

Manual DB cluster snapshot attributes are used to authorize other AWS accounts to copy or restore a manual DB cluster snapshot. For more information, see the ModifyDBClusterSnapshotAttribute (p. 257) API action.

Contents

DBClusterSnapshotAttributes.DBClusterSnapshotAttribute.N

The list of attributes and values for the manual DB cluster snapshot.

Type: Array of DBClusterSnapshotAttribute (p. 316) objects

Required: No

DBClusterSnapshotIdentifier

The identifier of the manual DB cluster snapshot that the attributes apply to.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
DBEngineVersion

This data type is used as a response element in the action DescribeDBEngineVersions (p. 217).

Contents

DBEngineDescription
   The description of the database engine.
   Type: String
   Required: No

DBEngineVersionDescription
   The description of the database engine version.
   Type: String
   Required: No

DBParameterGroupFamily
   The name of the DB parameter group family for the database engine.
   Type: String
   Required: No

DefaultCharacterSet
   The default character set for new instances of this engine version, if the CharacterSetName parameter of the CreateDBInstance API is not specified.
   Type: CharacterSet (p. 302) object
   Required: No

Engine
   The name of the database engine.
   Type: String
   Required: No

EngineVersion
   The version number of the database engine.
   Type: String
   Required: No

ExportableLogTypes.member.N
   The types of logs that the database engine has available for export to CloudWatch Logs.
   Type: Array of strings
   Required: No

SupportedCharacterSets.CharacterSet.N
   A list of the character sets supported by this engine for the CharacterSetName parameter of the CreateDBInstance action.
Type: Array of `CharacterSet (p. 302)` objects

Required: No

**SupportedTimezones.Timezone.N**

A list of the time zones supported by this engine for the `Timezone` parameter of the `CreateDBInstance` action.

Type: Array of `Timezone (p. 359)` objects

Required: No

**SupportsLogExportsToCloudwatchLogs**

A value that indicates whether the engine version supports exporting the log types specified by `ExportableLogTypes` to CloudWatch Logs.

Type: Boolean

Required: No

**SupportsReadReplica**

Indicates whether the database engine version supports read replicas.

Type: Boolean

Required: No

**ValidUpgradeTarget.UpgradeTarget.N**

A list of engine versions that this database engine version can be upgraded to.

Type: Array of `UpgradeTarget (p. 360)` objects

Required: No

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
DBInstance

Contains the details of an Amazon Neptune DB instance.
This data type is used as a response element in the DescribeDBInstances (p. 220) action.

Contents

AllocatedStorage

  Specifies the allocated storage size specified in gibibytes.
  Type: Integer
  Required: No

AutoMinorVersionUpgrade

  Indicates that minor version patches are applied automatically.
  Type: Boolean
  Required: No

AvailabilityZone

  Specifies the name of the Availability Zone the DB instance is located in.
  Type: String
  Required: No

BackupRetentionPeriod

  Specifies the number of days for which automatic DB snapshots are retained.
  Type: Integer
  Required: No

CACertificateIdentifier

  The identifier of the CA certificate for this DB instance.
  Type: String
  Required: No

CharacterSetName

  If present, specifies the name of the character set that this instance is associated with.
  Type: String
  Required: No

CopyTagsToSnapshot

  Specifies whether tags are copied from the DB instance to snapshots of the DB instance.
  Type: Boolean
  Required: No
**DBClusterIdentifier**

If the DB instance is a member of a DB cluster, contains the name of the DB cluster that the DB instance is a member of.

Type: String

Required: No

**DBInstanceArn**

The Amazon Resource Name (ARN) for the DB instance.

Type: String

Required: No

**DBInstanceClass**

Contains the name of the compute and memory capacity class of the DB instance.

Type: String

Required: No

**DBInstanceIdentifier**

Contains a user-supplied database identifier. This identifier is the unique key that identifies a DB instance.

Type: String

Required: No

**DBInstancePort**

Specifies the port that the DB instance listens on. If the DB instance is part of a DB cluster, this can be a different port than the DB cluster port.

Type: Integer

Required: No

**DBInstanceStatus**

Specifies the current state of this database.

Type: String

Required: No

**DbiResourceId**

The AWS Region-unique, immutable identifier for the DB instance. This identifier is found in AWS CloudTrail log entries whenever the AWS KMS key for the DB instance is accessed.

Type: String

Required: No

**DBName**

The database name.

Type: String
Required: No

**DBParameterGroups.DBParameterGroup.N**

Provides the list of DB parameter groups applied to this DB instance.

Type: Array of DBParameterGroupStatus (p. 329) objects

Required: No

**DBSecurityGroups.DBSecurityGroup.N**

Provides List of DB security group elements containing only DBSecurityGroup.Name and DBSecurityGroup.Status subelements.

Type: Array of DBSecurityGroupMembership (p. 330) objects

Required: No

**DBSubnetGroup**

Specifies information on the subnet group associated with the DB instance, including the name, description, and subnets in the subnet group.

Type: DBSubnetGroup (p. 331) object

Required: No

**DomainMemberships.DomainMembership.N**

Not supported

Type: Array of DomainMembership (p. 333) objects

Required: No

**EnabledCloudwatchLogsExports.member.N**

A list of log types that this DB instance is configured to export to CloudWatch Logs.

Type: Array of strings

Required: No

**Endpoint**

Specifies the connection endpoint.

Type: Endpoint (p. 335) object

Required: No

**Engine**

Provides the name of the database engine to be used for this DB instance.

Type: String

Required: No

**EngineVersion**

Indicates the database engine version.

Type: String

Required: No
EnhancedMonitoringResourceArn

The Amazon Resource Name (ARN) of the Amazon CloudWatch Logs log stream that receives the Enhanced Monitoring metrics data for the DB instance.

Type: String
Required: No

IAMDatabaseAuthenticationEnabled

True if AWS Identity and Access Management (IAM) authentication is enabled, and otherwise false.

Type: Boolean
Required: No

InstanceCreateTime

Provides the date and time the DB instance was created.

Type: Timestamp
Required: No

Iops

Specifies the Provisioned IOPS (I/O operations per second) value.

Type: Integer
Required: No

KmsKeyId

If StorageEncrypted is true, the AWS KMS key identifier for the encrypted DB instance.

Type: String
Required: No

LatestRestorableTime

Specifies the latest time to which a database can be restored with point-in-time restore.

Type: Timestamp
Required: No

LicenseModel

License model information for this DB instance.

Type: String
Required: No

MasterUsername

Contains the master username for the DB instance.

Type: String
Required: No

MonitoringInterval

The interval, in seconds, between points when Enhanced Monitoring metrics are collected for the DB instance.
**MonitoringRoleArn**

The ARN for the IAM role that permits Neptune to send Enhanced Monitoring metrics to Amazon CloudWatch Logs.

Type: String

Required: No

**MultiAZ**

Specifies if the DB instance is a Multi-AZ deployment.

Type: Boolean

Required: No

**OptionGroupMemberships.OptionGroupMembership.N**

Provides the list of option group memberships for this DB instance.

Type: Array of OptionGroupMembership (p. 343) objects

Required: No

**PendingModifiedValues**

Specifies that changes to the DB instance are pending. This element is only included when changes are pending. Specific changes are identified by subelements.

Type: PendingModifiedValues (p. 352) object

Required: No

**PerformanceInsightsEnabled**

True if Performance Insights is enabled for the DB instance, and otherwise false.

Type: Boolean

Required: No

**PerformanceInsightsKMSKeyId**

The AWS KMS key identifier for encryption of Performance Insights data. The KMS key ID is the Amazon Resource Name (ARN), KMS key identifier, or the KMS key alias for the KMS encryption key.

Type: String

Required: No

**PreferredBackupWindow**

Specifies the daily time range during which automated backups are created if automated backups are enabled, as determined by the BackupRetentionPeriod.

Type: String

Required: No

**PreferredMaintenanceWindow**

Specifies the weekly time range during which system maintenance can occur, in Universal Coordinated Time (UTC).
DBInstance

Type: String
Required: No

PromotionTier

A value that specifies the order in which a Read Replica is promoted to the primary instance after a failure of the existing primary instance.

Type: Integer
Required: No

PubliclyAccessible

This member has been deprecated.

This parameter is not supported.

Type: Boolean
Required: No

ReadReplicaDBClusterIdentifiers.ReadReplicaDBClusterIdentifier.N

Contains one or more identifiers of DB clusters that are Read Replicas of this DB instance.

Type: Array of strings
Required: No

ReadReplicaDBInstanceIdentifiers.ReadReplicaDBInstanceIdentifier.N

Contains one or more identifiers of the Read Replicas associated with this DB instance.

Type: Array of strings
Required: No

ReadReplicaSourceDBInstanceIdentifier

Contains the identifier of the source DB instance if this DB instance is a Read Replica.

Type: String
Required: No

SecondaryAvailabilityZone

If present, specifies the name of the secondary Availability Zone for a DB instance with multi-AZ support.

Type: String
Required: No

StatusInfos.DBInstanceStatusInfo.N

The status of a Read Replica. If the instance is not a Read Replica, this is blank.

Type: Array of DBInstanceStatusInfo (p. 327) objects
Required: No

StorageEncrypted

Specifies whether the DB instance is encrypted.
Type: Boolean
Required: No

**StorageType**

Specifies the storage type associated with DB instance.

Type: String
Required: No

**TdeCredentialArn**

The ARN from the key store with which the instance is associated for TDE encryption.

Type: String
Required: No

**Timezone**

Not supported.

Type: String
Required: No


Provides a list of VPC security group elements that the DB instance belongs to.

Type: Array of VpcSecurityGroupMembership (p. 363) objects
Required: No

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
DBInstanceStatusInfo

Provides a list of status information for a DB instance.

Contents

Message

Details of the error if there is an error for the instance. If the instance is not in an error state, this value is blank.

Type: String

Required: No

Normal

Boolean value that is true if the instance is operating normally, or false if the instance is in an error state.

Type: Boolean

Required: No

Status

Status of the DB instance. For a StatusType of read replica, the values can be replicating, error, stopped, or terminated.

Type: String

Required: No

StatusType

This value is currently "read replication."

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
**DBParameterGroup**

Contains the details of an Amazon Neptune DB parameter group.

This data type is used as a response element in the `DescribeDBParameterGroups` (p. 222) action.

**Contents**

**DBParameterGroupArn**

The Amazon Resource Name (ARN) for the DB parameter group.

Type: String

Required: No

**DBParameterGroupFamily**

Provides the name of the DB parameter group family that this DB parameter group is compatible with.

Type: String

Required: No

**DBParameterGroupName**

Provides the name of the DB parameter group.

Type: String

Required: No

**Description**

Provides the customer-specified description for this DB parameter group.

Type: String

Required: No

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
DBParameterGroupStatus

The status of the DB parameter group.

This data type is used as a response element in the following actions:

- CreateDBInstance (p. 175)
- DeleteDBInstance (p. 198)
- ModifyDBInstance (p. 260)
- RebootDBInstance (p. 278)

Contents

DBParameterGroupName

The name of the DB parameter group.

Type: String

Required: No

ParameterApplyStatus

The status of parameter updates.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
DBSecurityGroupMembership

This data type is used as a response element in the following actions:

- ModifyDBInstance (p. 260)
- RebootDBInstance (p. 278)

Contents

**DBSecurityGroupName**

The name of the DB security group.

Type: String

Required: No

**Status**

The status of the DB security group.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
DBSubnetGroup

Contains the details of an Amazon Neptune DB subnet group.

This data type is used as a response element in the DescribeDBSubnetGroups (p. 226) action.

Contents

DBSubnetGroupArn

The Amazon Resource Name (ARN) for the DB subnet group.

Type: String

Required: No

DBSubnetGroupDescription

Provides the description of the DB subnet group.

Type: String

Required: No

DBSubnetGroupName

The name of the DB subnet group.

Type: String

Required: No

SubnetGroupStatus

Provides the status of the DB subnet group.

Type: String

Required: No

Subnets.Subnet.N

Contains a list of Subnet (p. 357) elements.

Type: Array of Subnet (p. 357) objects

Required: No

VpcId

Provides the VpcId of the DB subnet group.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go

API Version 2017-11-29
- AWS SDK for Java
- AWS SDK for Ruby V2
DomainMembership

An Active Directory Domain membership record associated with the DB instance.

Contents

Domain

The identifier of the Active Directory Domain.

Type: String

Required: No

FQDN

The fully qualified domain name of the Active Directory Domain.

Type: String

Required: No

IAMRoleName

The name of the IAM role to be used when making API calls to the Directory Service.

Type: String

Required: No

Status

The status of the DB instance's Active Directory Domain membership, such as joined, pending-join, failed etc).

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
DoubleRange

A range of double values.

Contents

From

The minimum value in the range.

Type: Double

Required: No

To

The maximum value in the range.

Type: Double

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

• AWS SDK for C++
• AWS SDK for Go
• AWS SDK for Java
• AWS SDK for Ruby V2
Endpoint

This data type is used as a response element in the following actions:

- CreateDBInstance (p. 175)
- DescribeDBInstances (p. 220)
- DeleteDBInstance (p. 198)

Contents

Address

Specifies the DNS address of the DB instance.

Type: String

Required: No

HostedZoneId

Specifies the ID that Amazon Route 53 assigns when you create a hosted zone.

Type: String

Required: No

Port

Specifies the port that the database engine is listening on.

Type: Integer

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
EngineDefaults

Contains the result of a successful invocation of the DescribeEngineDefaultParameters (p. 230) action.

Contents

DBParameterGroupFamily

Specifies the name of the DB parameter group family that the engine default parameters apply to.

- Type: String
- Required: No

Marker

An optional pagination token provided by a previous EngineDefaults request. If this parameter is specified, the response includes only records beyond the marker, up to the value specified by MaxRecords.

- Type: String
- Required: No

Parameters.Parameter.N

Contains a list of engine default parameters.

- Type: Array of Parameter (p. 347) objects
- Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
Event

This data type is used as a response element in the DescribeEvents (p. 234) action.

Contents

Date
Specifies the date and time of the event.
Type: Timestamp
Required: No

EventCategories.EventCategory.N
Specifies the category for the event.
Type: Array of strings
Required: No

Message
Provides the text of this event.
Type: String
Required: No

SourceArn
The Amazon Resource Name (ARN) for the event.
Type: String
Required: No

SourceIdentifier
Provides the identifier for the source of the event.
Type: String
Required: No

SourceType
Specifies the source type for this event.
Type: String
Valid Values: db-instance | db-parameter-group | db-security-group | db-snapshot | db-cluster | db-cluster-snapshot
Required: No

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
EventCategoriesMap

Contains the results of a successful invocation of the DescribeEventCategories (p. 232) action.

Contents

**EventCategories.EventCategory.N**

The event categories for the specified source type

Type: Array of strings

Required: No

**SourceType**

The source type that the returned categories belong to

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
EventSubscription

Contains the results of a successful invocation of the DescribeEventSubscriptions (p. 237) action.

Contents

CustomerAwsId

The AWS customer account associated with the event notification subscription.

Type: String
Required: No

CustSubscriptionId

The event notification subscription Id.

Type: String
Required: No

Enabled

A Boolean value indicating if the subscription is enabled. True indicates the subscription is enabled.

Type: Boolean
Required: No

EventCategoriesList.EventCategory.N

A list of event categories for the event notification subscription.

Type: Array of strings
Required: No

EventSubscriptionArn

The Amazon Resource Name (ARN) for the event subscription.

Type: String
Required: No

SnsTopicArn

The topic ARN of the event notification subscription.

Type: String
Required: No

SourceIdsList.SourceId.N

A list of source IDs for the event notification subscription.

Type: Array of strings
Required: No

SourceType

The source type for the event notification subscription.
EventSubscription

Type: String
Required: No

**Status**

The status of the event notification subscription.

Constraints:

Can be one of the following: creating | modifying | deleting | active | no-permission | topic-not-exist

The status "no-permission" indicates that Neptune no longer has permission to post to the SNS topic. The status "topic-not-exist" indicates that the topic was deleted after the subscription was created.

Type: String
Required: No

**SubscriptionCreationTime**

The time the event notification subscription was created.

Type: String
Required: No

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
Filter

This type is not currently supported.

Contents

Name

This parameter is not currently supported.

Type: String

Required: Yes

Values.Value.N

This parameter is not currently supported.

Type: Array of strings

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
OptionGroupMembership

Provides information on the option groups the DB instance is a member of.

Contents

OptionGroupName

The name of the option group that the instance belongs to.

Type: String

Required: No

Status

The status of the DB instance's option group membership. Valid values are: in-sync, pending-apply, pending-removal, pending-maintenance-apply, pending-maintenance-removal, applying, removing, and failed.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
OrderableDBInstanceOption

Contains a list of available options for a DB instance.

This data type is used as a response element in the DescribeOrderableDBInstanceOptions (p. 239) action.

Contents

AvailabilityZones.AvailabilityZone.N

A list of Availability Zones for a DB instance.

Type: Array of AvailabilityZone (p. 301) objects

Required: No

DBInstanceClass

The DB instance class for a DB instance.

Type: String

Required: No

Engine

The engine type of a DB instance.

Type: String

Required: No

EngineVersion

The engine version of a DB instance.

Type: String

Required: No

LicenseModel

The license model for a DB instance.

Type: String

Required: No

MaxIopsPerDbInstance

Maximum total provisioned IOPS for a DB instance.

Type: Integer

Required: No

MaxIopsPerGib

Maximum provisioned IOPS per GiB for a DB instance.

Type: Double

Required: No
MaxStorageSize

Maximum storage size for a DB instance.
Type: Integer
Required: No

MinIopsPerDbInstance

Minimum total provisioned IOPS for a DB instance.
Type: Integer
Required: No

MinIopsPerGib

Minimum provisioned IOPS per GiB for a DB instance.
Type: Double
Required: No

MinStorageSize

Minimum storage size for a DB instance.
Type: Integer
Required: No

MultiAZCapable

Indicates whether a DB instance is Multi-AZ capable.
Type: Boolean
Required: No

ReadReplicaCapable

Indicates whether a DB instance can have a Read Replica.
Type: Boolean
Required: No

StorageType

Indicates the storage type for a DB instance.
Type: String
Required: No

SupportsEnhancedMonitoring

Indicates whether a DB instance supports Enhanced Monitoring at intervals from 1 to 60 seconds.
Type: Boolean
Required: No

SupportsIAMDatabaseAuthentication

Indicates whether a DB instance supports IAM database authentication.
Type: Boolean
Required: No

**SupportsIops**

Indicates whether a DB instance supports provisioned IOPS.

Type: Boolean
Required: No

**SupportsPerformanceInsights**

True if a DB instance supports Performance Insights, otherwise false.

Type: Boolean
Required: No

**SupportsStorageEncryption**

Indicates whether a DB instance supports encrypted storage.

Type: Boolean
Required: No

**Vpc**

Indicates whether a DB instance is in a VPC.

Type: Boolean
Required: No

### See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
Parameter

This data type is used as a request parameter in the ModifyDBParameterGroup (p. 270) and ResetDBParameterGroup (p. 288) actions.

This data type is used as a response element in the DescribeEngineDefaultParameters (p. 230) and DescribeDBParameters (p. 224) actions.

Contents

AllowedValues

Specifies the valid range of values for the parameter.

- Type: String
- Required: No

ApplyMethod

Indicates when to apply parameter updates.

- Type: String
- Valid Values: immediate | pending-reboot
- Required: No

ApplyType

Specifies the engine specific parameters type.

- Type: String
- Required: No

DataType

Specifies the valid data type for the parameter.

- Type: String
- Required: No

Description

Provides a description of the parameter.

- Type: String
- Required: No

IsModifiable

Indicates whether (true) or not (false) the parameter can be modified. Some parameters have security or operational implications that prevent them from being changed.

- Type: Boolean
- Required: No

MinimumEngineVersion

The earliest engine version to which the parameter can apply.
Parameter

Type: String
Required: No

ParameterName

Specifies the name of the parameter.
Type: String
Required: No

ParameterValue

Specifies the value of the parameter.
Type: String
Required: No

Source

Indicates the source of the parameter value.
Type: String
Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
PendingCloudwatchLogsExports

A list of the log types whose configuration is still pending. In other words, these log types are in the process of being activated or deactivated.

Contents

LogTypesToDisable.member.N
Log types that are in the process of being enabled. After they are enabled, these log types are exported to CloudWatch Logs.

Type: Array of strings
Required: No

LogTypesToEnable.member.N
Log types that are in the process of being deactivated. After they are deactivated, these log types aren't exported to CloudWatch Logs.

Type: Array of strings
Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
PendingMaintenanceAction

Provides information about a pending maintenance action for a resource.

Contents

**Action**

The type of pending maintenance action that is available for the resource.

Type: String

Required: No

**AutoAppliedAfterDate**

The date of the maintenance window when the action is applied. The maintenance action is applied to the resource during its first maintenance window after this date. If this date is specified, any next-maintenance opt-in requests are ignored.

Type: Timestamp

Required: No

**CurrentApplyDate**

The effective date when the pending maintenance action is applied to the resource. This date takes into account opt-in requests received from the ApplyPendingMaintenanceAction (p. 154) API, the AutoAppliedAfterDate, and the ForcedApplyDate. This value is blank if an opt-in request has not been received and nothing has been specified as AutoAppliedAfterDate or ForcedApplyDate.

Type: Timestamp

Required: No

**Description**

A description providing more detail about the maintenance action.

Type: String

Required: No

**ForcedApplyDate**

The date when the maintenance action is automatically applied. The maintenance action is applied to the resource on this date regardless of the maintenance window for the resource. If this date is specified, any immediate opt-in requests are ignored.

Type: Timestamp

Required: No

**OptInStatus**

Indicates the type of opt-in request that has been received for the resource.

Type: String

Required: No
See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
PendingModifiedValues

This data type is used as a response element in the ModifyDBInstance (p. 260) action.

Contents

AllocatedStorage

Contains the new AllocatedStorage size for the DB instance that will be applied or is currently being applied.

Type: Integer

Required: No

BackupRetentionPeriod

Specifies the pending number of days for which automated backups are retained.

Type: Integer

Required: No

CACertificateIdentifier

Specifies the identifier of the CA certificate for the DB instance.

Type: String

Required: No

DBInstanceClass

Contains the new DBInstanceClass for the DB instance that will be applied or is currently being applied.

Type: String

Required: No

DBInstanceIdentifier

Contains the new DBInstanceIdentifier for the DB instance that will be applied or is currently being applied.

Type: String

Required: No

DBSubnetGroupName

The new DB subnet group for the DB instance.

Type: String

Required: No

EngineVersion

Indicates the database engine version.

Type: String

Required: No
**Iops**

Specifies the new Provisioned IOPS value for the DB instance that will be applied or is currently being applied.

Type: Integer

Required: No

**LicenseModel**

The license model for the DB instance.

Valid values: license-included | bring-your-own-license | general-public-license

Type: String

Required: No

**MasterUserPassword**

Contains the pending or currently-in-progress change of the master credentials for the DB instance.

Type: String

Required: No

**MultiAZ**

Indicates that the Single-AZ DB instance is to change to a Multi-AZ deployment.

Type: Boolean

Required: No

**PendingCloudwatchLogsExports**

A list of the log types whose configuration is still pending. In other words, these log types are in the process of being activated or deactivated.

Type: `PendingCloudwatchLogsExports (p. 349)` object

Required: No

**Port**

Specifies the pending port for the DB instance.

Type: Integer

Required: No

**StorageType**

Specifies the storage type to be associated with the DB instance.

Type: String

Required: No

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
• AWS SDK for Go
• AWS SDK for Java
• AWS SDK for Ruby V2
Range

A range of integer values.

Contents

From

The minimum value in the range.

Type: Integer

Required: No

Step

The step value for the range. For example, if you have a range of 5,000 to 10,000, with a step value of 1,000, the valid values start at 5,000 and step up by 1,000. Even though 7,500 is within the range, it isn't a valid value for the range. The valid values are 5,000, 6,000, 7,000, 8,000...

Type: Integer

Required: No

To

The maximum value in the range.

Type: Integer

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
ResourcePendingMaintenanceActions

Describes the pending maintenance actions for a resource.

Contents

PendingMaintenanceActionDetails.PendingMaintenanceAction.N

A list that provides details about the pending maintenance actions for the resource.

Type: Array of PendingMaintenanceAction (p. 350) objects

Required: No

ResourceIdentifier

The ARN of the resource that has pending maintenance actions.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
Subnet

This data type is used as a response element in the DescribeDBSubnetGroups (p. 226) action.

Contents

SubnetAvailabilityZone

Contains Availability Zone information.

This data type is used as an element in the following data type:

- OrderableDBInstanceOption (p. 344)

Type: AvailabilityZone (p. 301) object

Required: No

SubnetIdentifier

Specifies the identifier of the subnet.

Type: String

Required: No

SubnetStatus

Specifies the status of the subnet.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
Tag

Metadata assigned to an Amazon Neptune resource consisting of a key-value pair.

Contents

Key

A key is the required name of the tag. The string value can be from 1 to 128 Unicode characters in length and can't be prefixed with "aws:” or "rds:“. The string can only contain only the set of Unicode letters, digits, white-space, ',', ';', '/', '=', '+', '-' (Java regex: "^[\p{L}\p{Z}\p{N}_.:/=+\-]+$").

Type: String
Required: No

Value

A value is the optional value of the tag. The string value can be from 1 to 256 Unicode characters in length and can't be prefixed with "aws:” or "rds:“. The string can only contain only the set of Unicode letters, digits, white-space, ',', ';', '/', '=', '+', '-' (Java regex: "^[\p{L}\p{Z}\p{N}_.:/=+\-]+$").

Type: String
Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
Timezone

A time zone associated with a DBInstance (p. 320). This data type is an element in the response to the DescribeDBInstances (p. 220), and the DescribeDBEngineVersions (p. 217) actions.

Contents

TimezoneName

  The name of the time zone.

  Type: String

  Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
UpgradeTarget

The version of the database engine that a DB instance can be upgraded to.

Contents

AutoUpgrade

A value that indicates whether the target version is applied to any source DB instances that have AutoMinorVersionUpgrade set to true.

Type: Boolean
Required: No

Description

The version of the database engine that a DB instance can be upgraded to.

Type: String
Required: No

Engine

The name of the upgrade target database engine.

Type: String
Required: No

EngineVersion

The version number of the upgrade target database engine.

Type: String
Required: No

IsMajorVersionUpgrade

A value that indicates whether a database engine is upgraded to a major version.

Type: Boolean
Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
ValidDBInstanceModificationsMessage

Information about valid modifications that you can make to your DB instance. Contains the result of a successful call to the DescribeValidDBInstanceModifications (p. 244) action. You can use this information when you call ModifyDBInstance (p. 260).

Contents

Storage.ValidStorageOptions.N

Valid storage options for your DB instance.

Type: Array of ValidStorageOptions (p. 362) objects

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
ValidStorageOptions

Information about valid modifications that you can make to your DB instance. Contains the result of a successful call to the DescribeValidDBInstanceModifications (p. 244) action.

Contents

**lopsToStorageRatio.DoubleRange.N**

The valid range of Provisioned IOPS to gibibytes of storage multiplier. For example, 3-10, which means that provisioned IOPS can be between 3 and 10 times storage.

Type: Array of DoubleRange (p. 334) objects

Required: No

**ProvisionedIops.Range.N**

The valid range of provisioned IOPS. For example, 1000-20000.

Type: Array of Range (p. 355) objects

Required: No

**StorageSize.Range.N**

The valid range of storage in gibibytes. For example, 100 to 16384.

Type: Array of Range (p. 355) objects

Required: No

**StorageType**

The valid storage types for your DB instance. For example, gp2, io1.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
VpcSecurityGroupMembership
This data type is used as a response element for queries on VPC security group membership.

Contents

Status
The status of the VPC security group.
Type: String
Required: No

VpcSecurityGroupId
The name of the VPC security group.
Type: String
Required: No

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

Common Errors
This section lists the errors common to the API actions of all AWS services. For errors specific to an API action for this service, see the topic for that API action.

AccessDeniedException
You do not have sufficient access to perform this action.

HTTP Status Code: 400

IncompleteSignature
The request signature does not conform to AWS standards.

HTTP Status Code: 400

InternalFailure
The request processing has failed because of an unknown error, exception or failure.

HTTP Status Code: 500

InvalidAction
The action or operation requested is invalid. Verify that the action is typed correctly.
HTTP Status Code: 400
InvalidClientTokenid
The X.509 certificate or AWS access key ID provided does not exist in our records.

HTTP Status Code: 403
InvalidParameterCombination
Parameters that must not be used together were used together.

HTTP Status Code: 400
InvalidParameterValue
An invalid or out-of-range value was supplied for the input parameter.

HTTP Status Code: 400
InvalidQueryParameter
The AWS query string is malformed or does not adhere to AWS standards.

HTTP Status Code: 400
MalformedQueryString
The query string contains a syntax error.

HTTP Status Code: 404
MissingAction
The request is missing an action or a required parameter.

HTTP Status Code: 400
MissingAuthenticationToken
The request must contain either a valid (registered) AWS access key ID or X.509 certificate.

HTTP Status Code: 403
MissingParameter
A required parameter for the specified action is not supplied.

HTTP Status Code: 403
OptInRequired
The AWS access key ID needs a subscription for the service.

HTTP Status Code: 403
RequestExpired
The request reached the service more than 15 minutes after the date stamp on the request or more than 15 minutes after the request expiration date (such as for pre-signed URLs), or the date stamp on the request is more than 15 minutes in the future.

HTTP Status Code: 400
ServiceUnavailable
The request has failed due to a temporary failure of the server.

HTTP Status Code: 503
Common Parameters

The following list contains the parameters that all actions use for signing Signature Version 4 requests with a query string. Any action-specific parameters are listed in the topic for that action. For more information about Signature Version 4, see Signature Version 4 Signing Process in the Amazon Web Services General Reference.

**Action**

The action to be performed.

Type: string

Required: Yes

**Version**

The API version that the request is written for, expressed in the format YYYY-MM-DD.

Type: string

Required: Yes

**X-Amz-Algorithm**

The hash algorithm that you used to create the request signature.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Valid Values: AWS4-HMAC-SHA256

Required: Conditional

**X-Amz-Credential**

The credential scope value, which is a string that includes your access key, the date, the region you are targeting, the service you are requesting, and a termination string ("aws4_request"). The value is expressed in the following format: access_key/YYYYMMDD/region/service/aws4_request.

For more information, see Task 2: Create a String to Sign for Signature Version 4 in the Amazon Web Services General Reference.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string
Required: Conditional

**X-Amz-Date**

The date that is used to create the signature. The format must be ISO 8601 basic format (YYYYMMDD'T'HHMMSS'Z'). For example, the following date time is a valid X-Amz-Date value: 20120325T120000Z.

Condition: X-Amz-Date is optional for all requests; it can be used to override the date used for signing requests. If the Date header is specified in the ISO 8601 basic format, X-Amz-Date is not required. When X-Amz-Date is used, it always overrides the value of the Date header. For more information, see *Handling Dates in Signature Version 4* in the *Amazon Web Services General Reference*.

Type: string

Required: Conditional

**X-Amz-Security-Token**

The temporary security token that was obtained through a call to AWS Security Token Service (AWS STS). For a list of services that support temporary security credentials from AWS Security Token Service, go to *AWS Services That Work with IAM* in the *IAM User Guide*.

Condition: If you're using temporary security credentials from the AWS Security Token Service, you must include the security token.

Type: string

Required: Conditional

**X-Amz-Signature**

Specifies the hex-encoded signature that was calculated from the string to sign and the derived signing key.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

**X-Amz-SignedHeaders**

Specifies all the HTTP headers that were included as part of the canonical request. For more information about specifying signed headers, see *Task 1: Create a Canonical Request For Signature Version 4* in the *Amazon Web Services General Reference*.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional