NRD2 Data FeedをAWS S3にインポートするには

2023年10月4日

本文書では、Lambda関数を活用してWhoisXML APIが提供するNRD2 Data FeedをAWS S3バケットにダウンロードする基本的な方法を紹介します。AWS Lambdaは、サーバーをプロビジョニングしたり管理したりすることなくコードを書いて実行できる、サーバーレスのコンピューティングサービスです。AWS S3は、ファイルを保存および取得するためのオブジェクトストレージサービスです。ここでは、AWS LambdaとAWS S3バケットの両方を構成するプロセスを説明します。

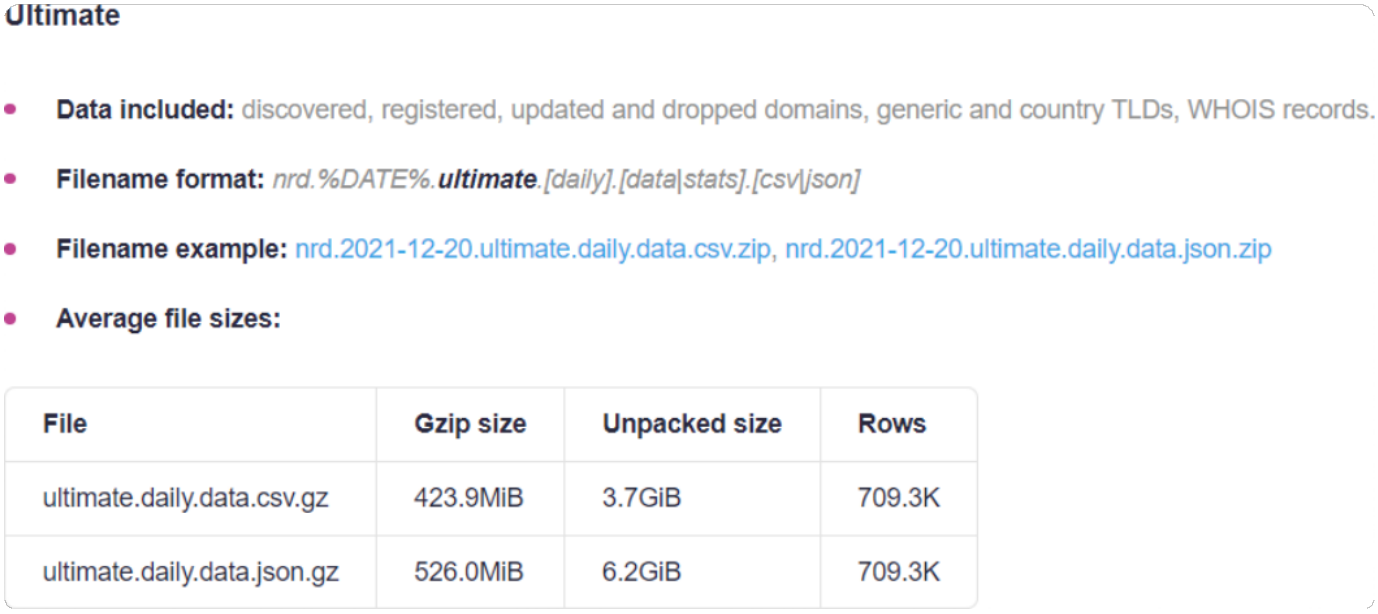
以下は本文書の対象外です：

- Lambda関数のスケジューリング
- ETLパイプライン
- PythonのRequestsモジュールのインポート
- 高度なセキュリティ
- クリーンアップ、ライフサイクルファイル管理

前提条件

事前に以下を用意する必要があります：

- AWSアカウント
- AWSサービス、特にAWS LambdaとS3に関する基礎～中級程度の知識
- Lambda関数で使われるPythonの知識
- WhoisXML APIのNRD2 Data Feedへのアクセス。この例では、NRD2 Ultimate:Simpleファイルを使用します。APIキーが必要です。詳細につきましては、sales@whoisxmlapi.com にお問い合わせください。NRD2の仕様は、こちらでご確認いただけます。



Ultimate

- **Data included:** discovered, registered, updated and dropped domains, generic and country TLDs, WHOIS records.
- **Filename format:** *nrd.%DATE%.**ultimate**.[daily].[data|stats].[csv|json]*
- **Filename example:** nrd.2021-12-20.ultimate.daily.data.csv.zip, nrd.2021-12-20.ultimate.daily.data.json.zip
- **Average file sizes:**

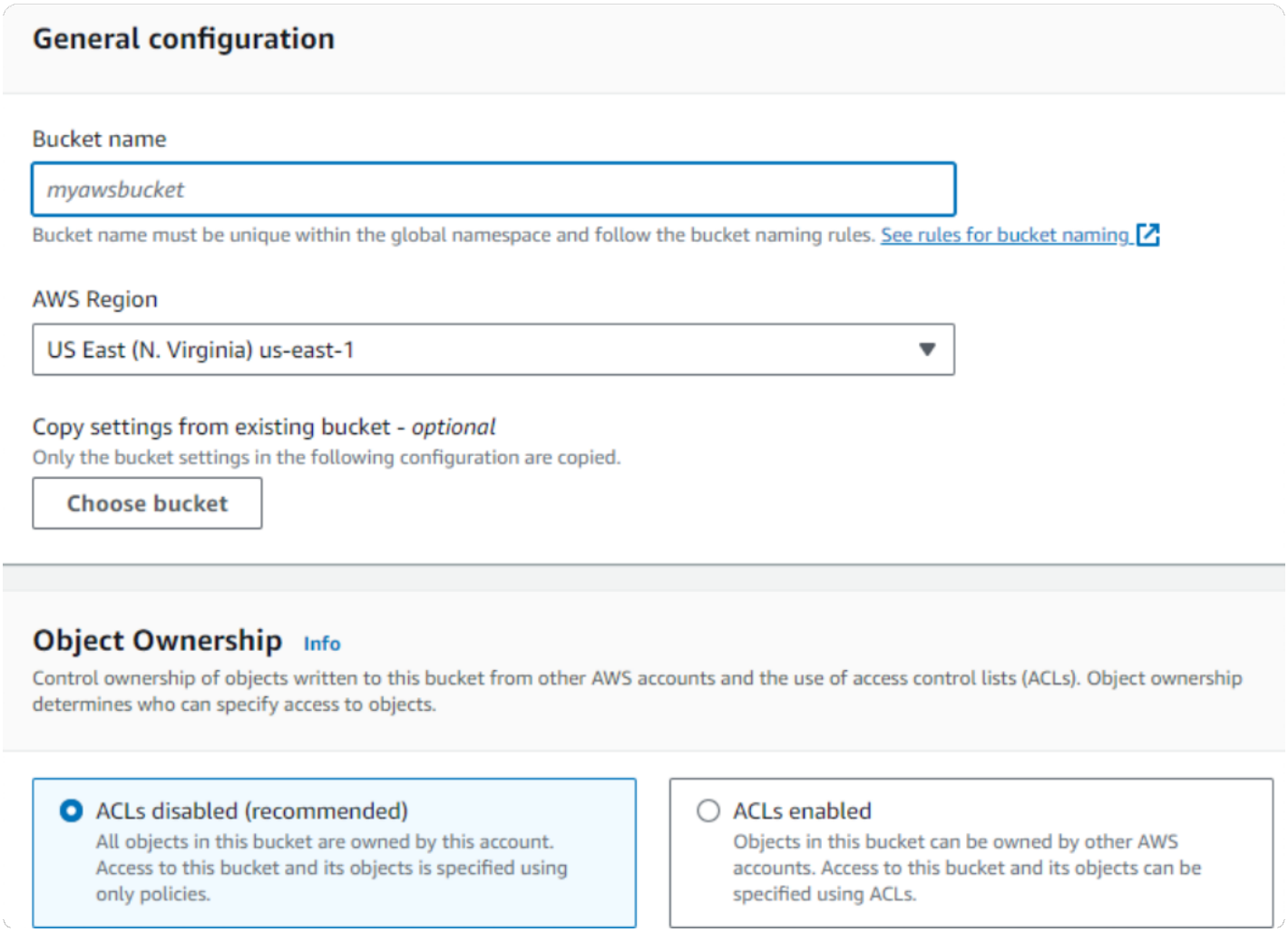
File	Gzip size	Unpacked size	Rows
ultimate.daily.data.csv.gz	423.9MiB	3.7GiB	709.3K
ultimate.daily.data.json.gz	526.0MiB	6.2GiB	709.3K

ステップ1：AWS S3バケットの作成

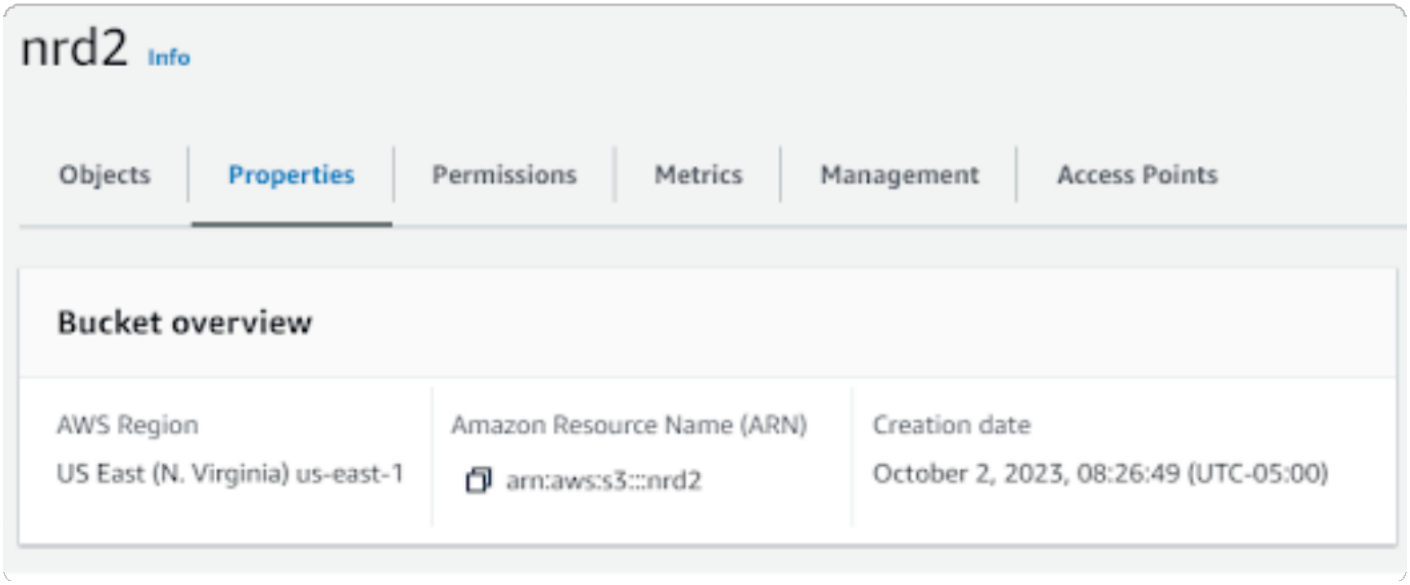
最初のステップは、NRD2ファイルを書き込むS3バケットの作成です。

- AWS Management Consoleで、S3サービスに移動します。
- 「Create Bucket」をクリックします。

- バケットにユニークな名前をつけ、適切な地域を選択します。



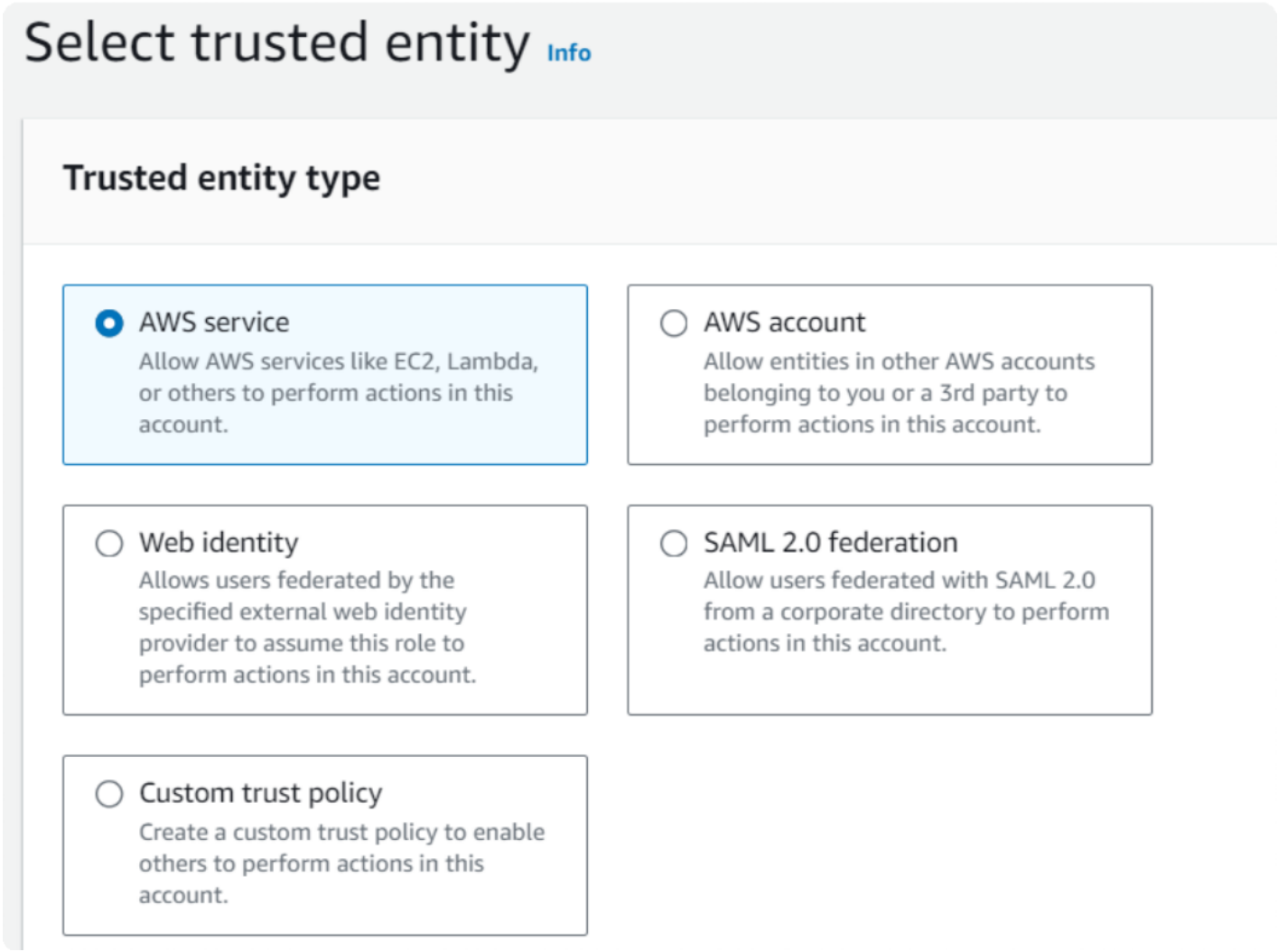
- ここでは、デフォルト設定のまま「Create Bucket」をクリックします。



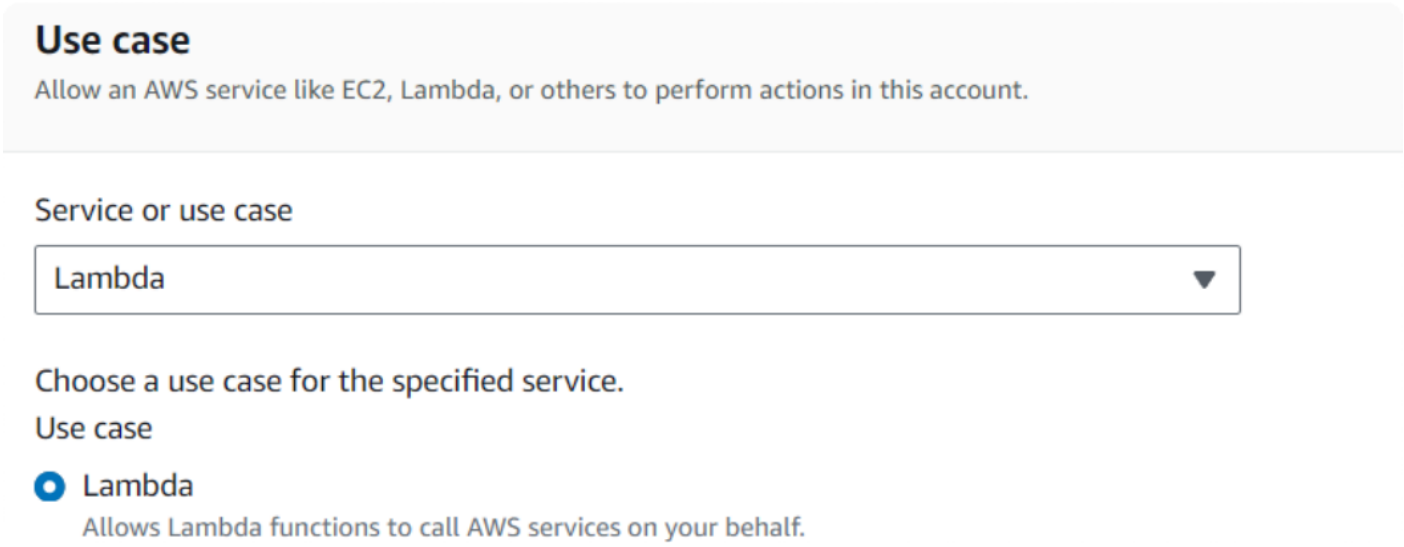
ステップ2： IAMロールの作成

AWS Lambdaでは、S3バケットの読み書きに必要な権限を持つIAMロールが必須となります。以下の手順でIAMロールを作成してください：

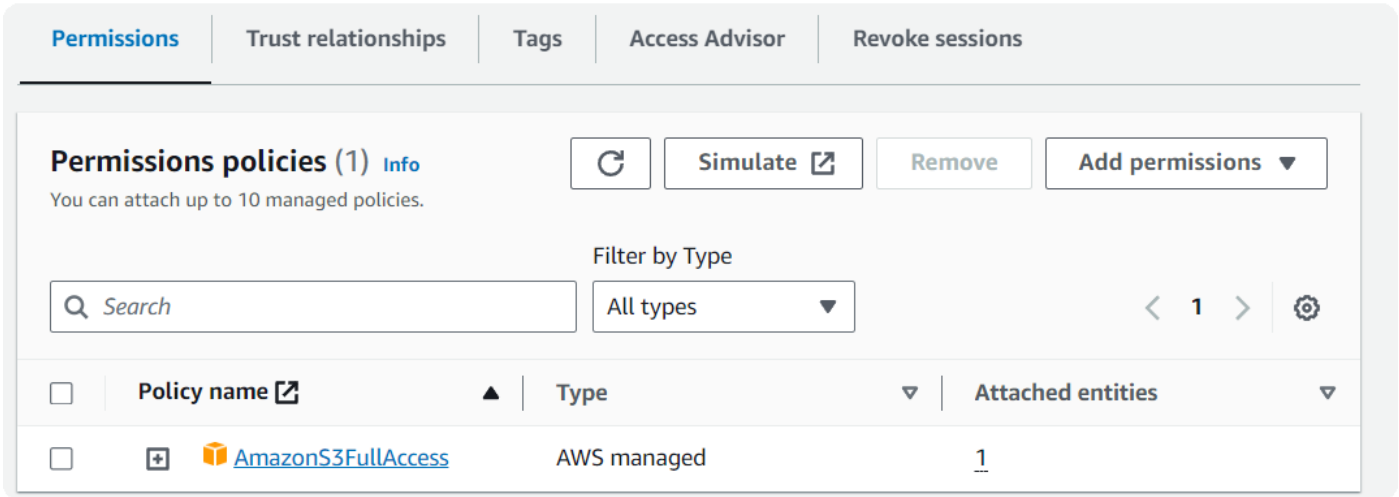
- AWSマネジメントコンソールでIAMサービスに移動します。



- 「Roles」をクリックし、次に「Create Role」をクリックします。
- このロールのサービスとして「Lambda」を選択し、「Next: Permissions」をクリックします。



- 検索バーに「S3」と入力し、「AWSS3FullAccess」、そして「Next: Tags」を選択します。



- タグは任意です。次に「Next: Review」をクリックします。
- あなたのロール、名前、簡単な説明を入力し、「Create Role」をクリックします。

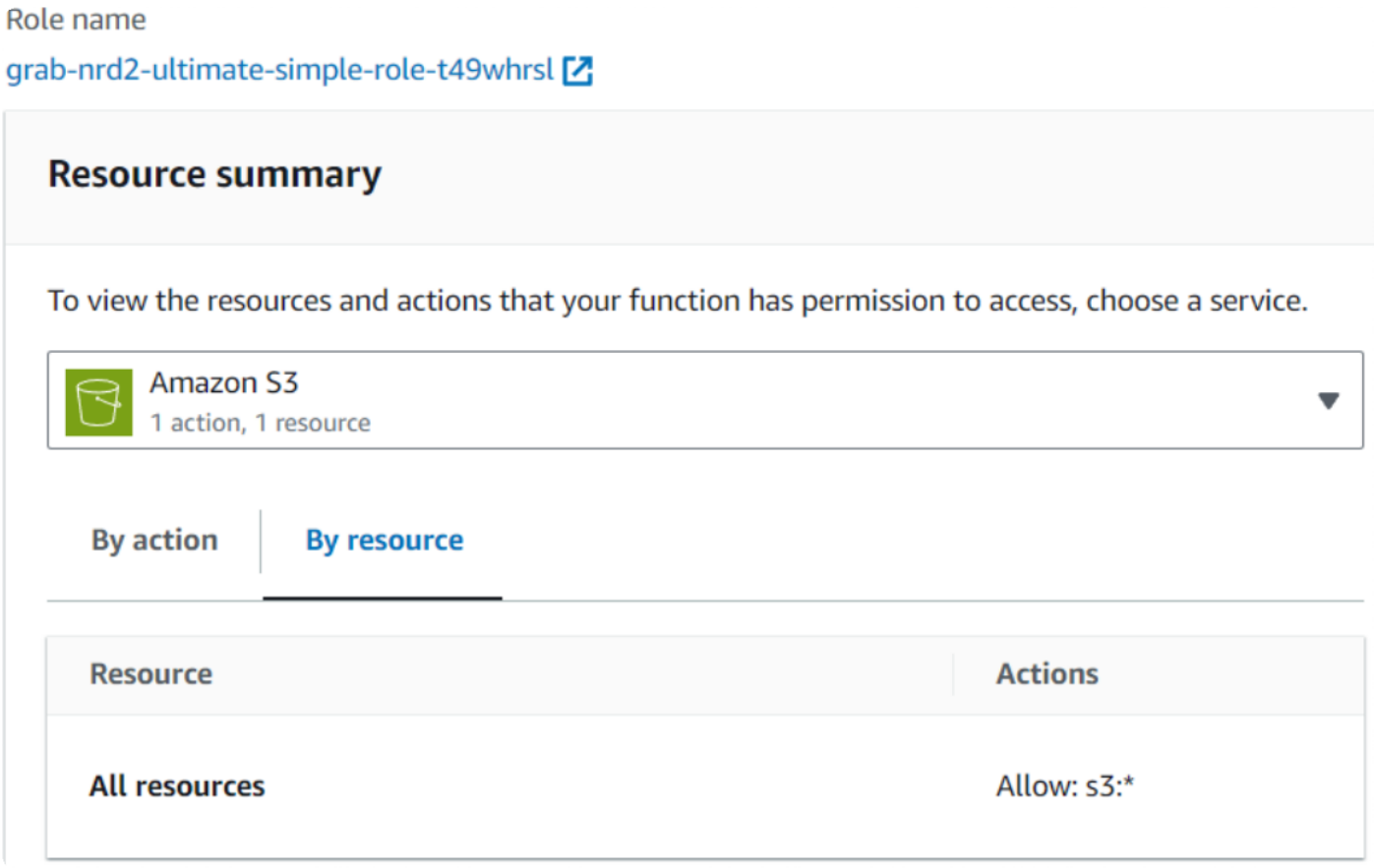
ステップ3：Lambda関数の作成

Lambda関数の作成は楽しく、簡単です。その方法は以下の通りです：

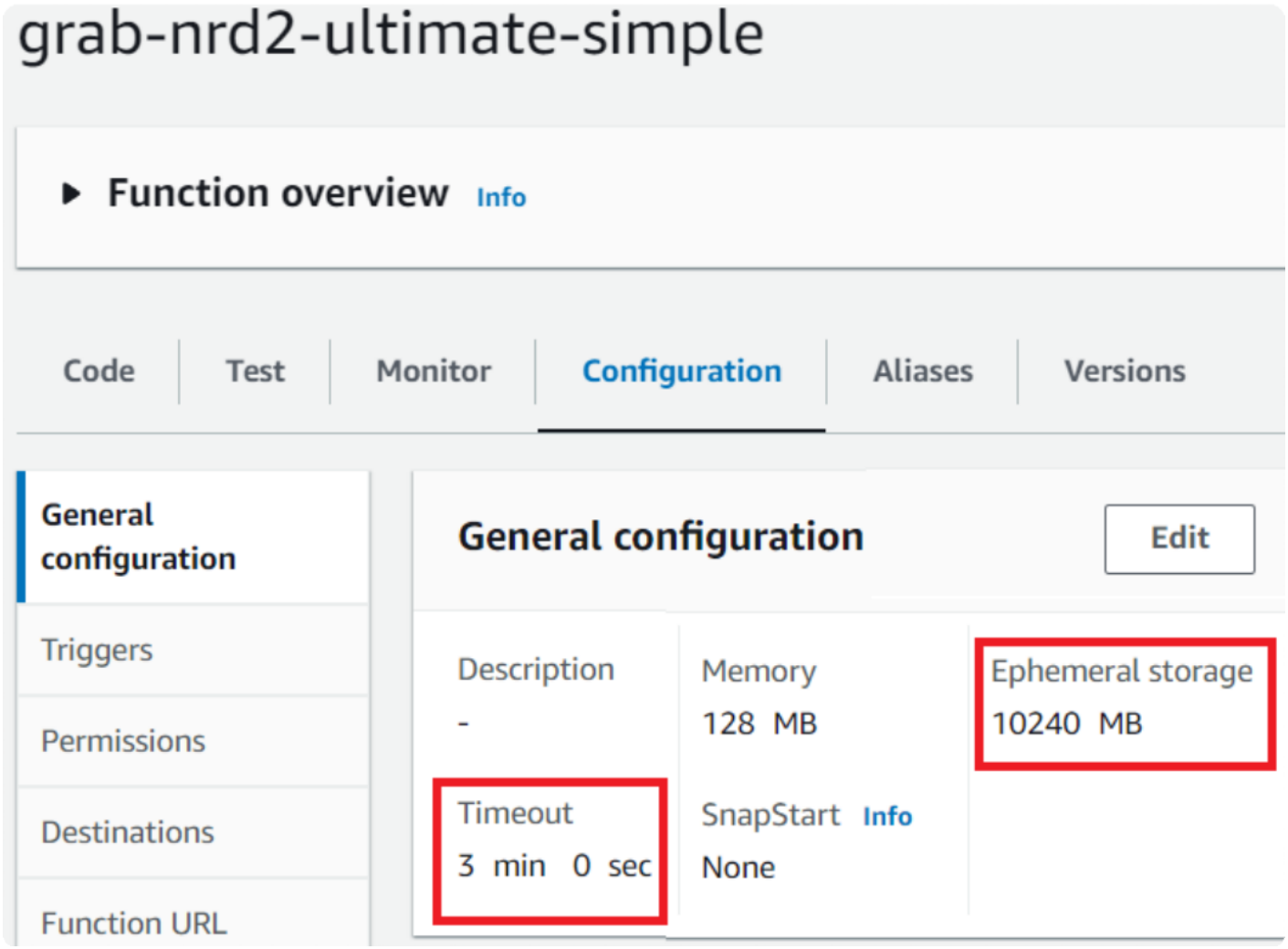
- AWSマネジメントコンソールでLambdaサービスに移動します。
- 「Create Function」をクリックします。
- 関数にわかりやすい名前を付け、ランタイムとしてPythonを選択します。そして、上記のステップ2で作成したIAMロールを選択します。
- 「Create function」をクリックします。

注：

Execution roleの設定：



Lambda関数のタイムアウト値を設定します。今回は3分に設定しています。



ステップ4：NRD2 .csvファイルをS3にインポートするLambda関数を記述

この例ではpython requestsモジュールを使用していますが、Boto3の一部ではなくなったため、インポートする必要があるかもしれません。この方法に関するAWSのドキュメントは曖昧ですが、インターネット上で様々な技術記事を見つけることができます。

コードの例：

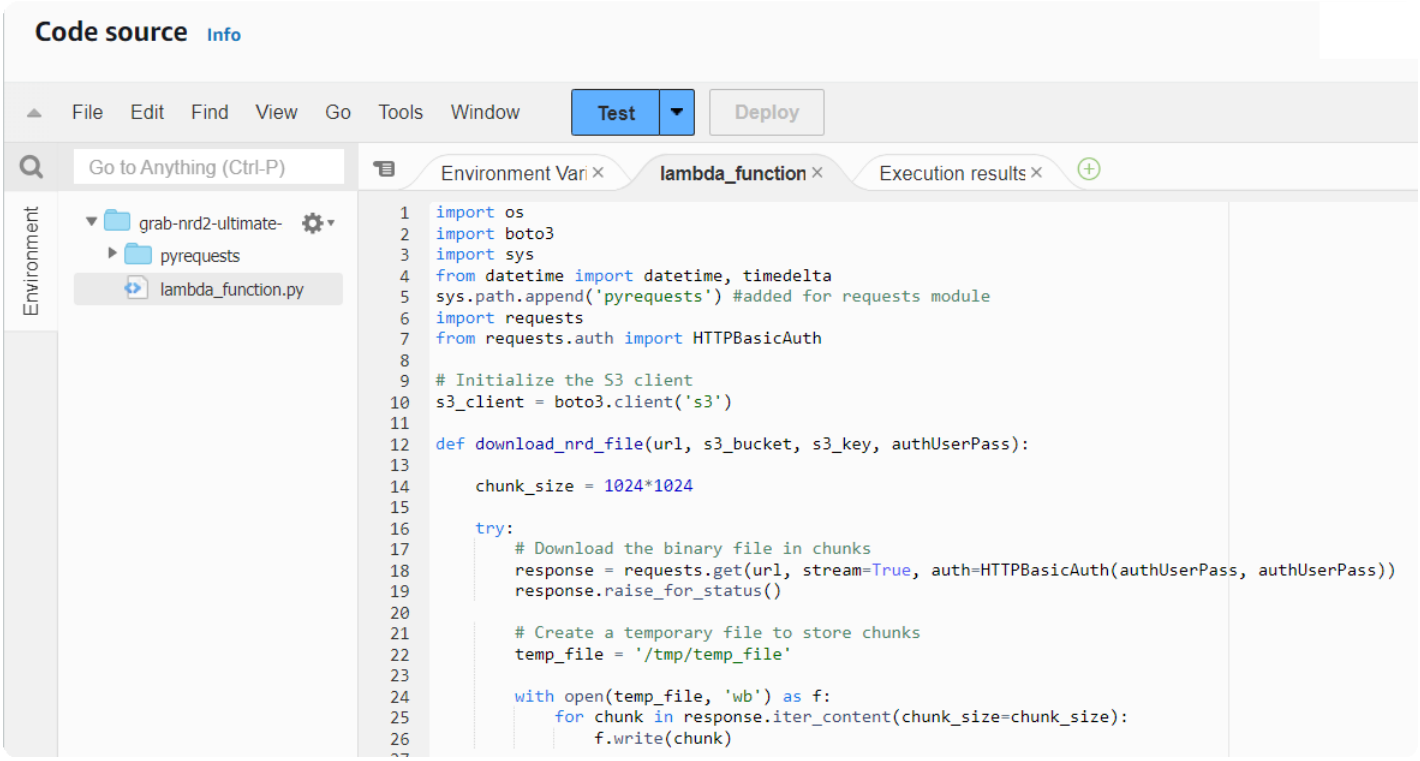
以下のPythonコードは、lambda_handlerのエントリーポイントです：

```
import os
import boto3
import sys
from datetime import datetime, timedelta
sys.path.append('pyrequests') #added for requests module
import requests
from requests.auth import HTTPBasicAuth

# Initialize the S3 client
s3_client = boto3.client('s3')

def download_nrd_file(url, s3_bucket, s3_key, authUserPass):

    chunk_size = 1024*1024

    try:
        # Download the binary file in chunks
        response = requests.get(url, stream=True, auth=HTTPBasicAuth(authUs
        response.raise_for_status()

        # Create a temporary file to store chunks
        temp_file = '/tmp/temp_file'

        with open(temp_file, 'wb') as f:
            for chunk in response.iter_content(chunk_size=chunk_size):
                f.write(chunk)

        # Upload the binary file to S3 from the temporary file
        s3_client.upload_file(temp_file, s3_bucket, s3_key)

        # Clean up the temporary file
        os.remove(temp_file)
```

```
            return True
        except Exception as e:
            print(f'Error: {str(e)}')
            return False


def lambda_handler(event, context):
    # Calculate yesterday's date in YYYY-MM-DD format
    yesterday = (datetime.now() - timedelta(days=1)).strftime("%Y-%m-%d")

    # Define the URL of the CSV file you want to download
    nrd_url = f"https://newly-registered-domains.whoisxmlapi.com/datafeeds/

    # Define your API Key here
    apiKey = "<YOUR_API_KEY"

    # Define the S3 bucket and object/key where you want to store the file
    s3_bucket = "nrd2"
    s3_key = f"nrd2-simple-{yesterday}.csv.gz"

    try:
        # Download the NRD2 file with basic authentication
        success = download_nrd_file(nrd_url, s3_bucket, s3_key, apiKey)

        print("Status code returned is ", str(success))

        if success:
            # Upload the NRD file to S3
            print(f"Uploading file to ", s3_bucket, s3_key)
            return {
                'statusCode': 200,
                'body': 'NRD2 file successfully downloaded and stored in S3
            }
        else:
            bodyStr = f"Failed to download {nrd_url}"
            return {
                'statusCode': 500,
                'body': bodyStr
            }
    except Exception as e:
```

```
        return {
            'statusCode': 500,
            'body': str(e)
        }
```

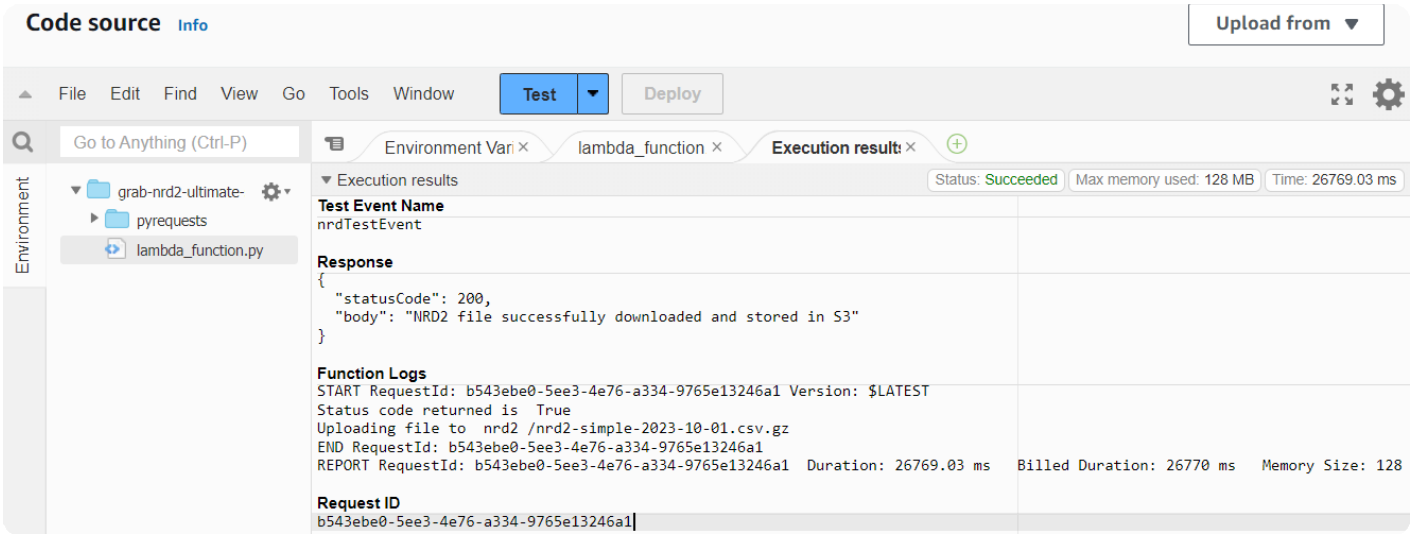
完成すると、このようなものができるはずです：



ステップ5：作成したLambda関数をテストする

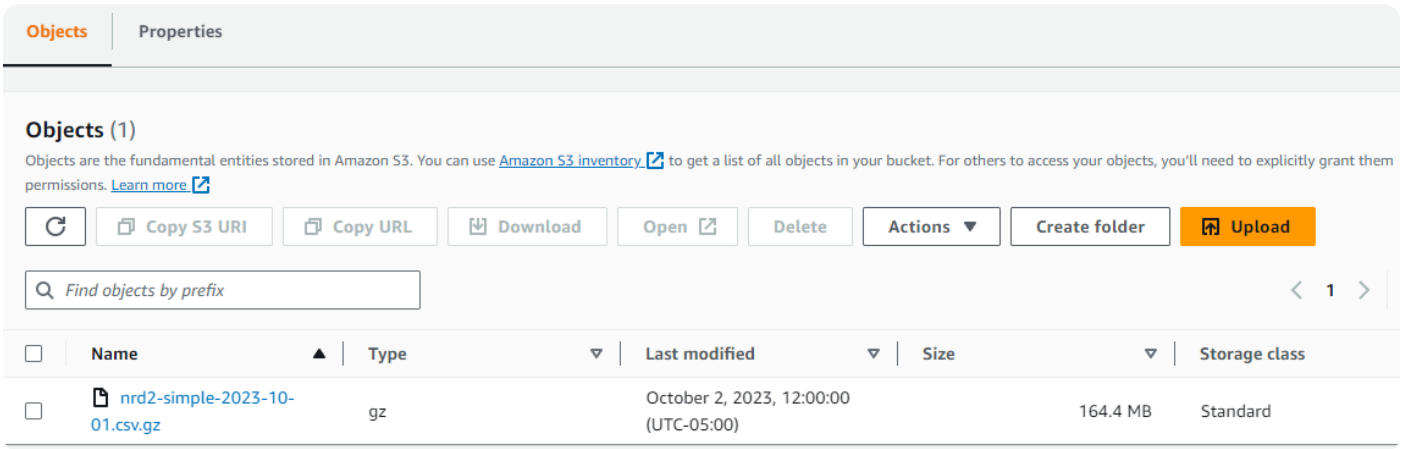
最後のステップとして、作成したLambda関数をテストし、a) NRD2ファイルを正常に取得できること、b) S3バケットに書き込めることを確認します：

- ページ上部の「Test」をクリックすると、以下のようなものが表示されるはずです。



- 「”requests" module not found 」というメッセージを受け取った場合は、[python requests library](#)を正しく設定する必要があります（本文書の範囲外）。

Lambda関数が正しく設定されていれば、関数はファイルを取得し、S3バケットに書き込みます。S3バケットに移動してファイルの存在を確認できます。



まとめ

S3バケットにアクセスできるAWS Lambdaの設定は、クラウドエンジニアにとってはごく一般的なタスクです。このプロセスの後に踏む次のステップは、Athena、PostgresまたはMySQLデータベースへのインポートなど、このデータで何をするかを決めることです。ETL用のAWS Glueをご存じない方は、そちらもチェックしてみてください。