L3 Exercise 2 - IaC

July 4, 2021

1 Exercise 2: Creating Redshift Cluster using the AWS python SDK

1.1 An example of Infrastructure-as-code

```
In [67]: import pandas as pd
         import boto3
         import json
```

2 STEP 0: Make sure you have an AWS secret and access key

- Create a new IAM user in your AWS account
- Give it `AdministratorAccess`, From `Attach existing policies directly` Tab
- Take note of the access key and secret
- Edit the file `dwh.cfg` in the same folder as this notebook and fill [AWS] KEY= YOUR_AWS_KEY SECRET= YOUR_AWS_SECRET

3 Load DWH Params from a file

```
In [68]: import configparser
         config = configparser.ConfigParser()
         config.read_file(open('dwh.cfg'))

         KEY                    = config.get('AWS','KEY')
         SECRET                 = config.get('AWS','SECRET')

         DWH_CLUSTER_TYPE       = config.get("DWH","DWH_CLUSTER_TYPE")
         DWH_NUM_NODES          = config.get("DWH","DWH_NUM_NODES")
         DWH_NODE_TYPE          = config.get("DWH","DWH_NODE_TYPE")

         DWH_CLUSTER_IDENTIFIER = config.get("DWH","DWH_CLUSTER_IDENTIFIER")
         DWH_DB                 = config.get("DWH","DWH_DB")
         DWH_DB_USER            = config.get("DWH","DWH_DB_USER")
         DWH_DB_PASSWORD        = config.get("DWH","DWH_DB_PASSWORD")
         DWH_PORT               = config.get("DWH","DWH_PORT")

         DWH_IAM_ROLE_NAME      = config.get("DWH", "DWH_IAM_ROLE_NAME")
```

```
        (DWH_DB_USER, DWH_DB_PASSWORD, DWH_DB)

        pd.DataFrame({"Param":
                          ["DWH_CLUSTER_TYPE", "DWH_NUM_NODES", "DWH_NODE_TYPE", "DWH_CLUSTER_I
                      "Value":
                          [DWH_CLUSTER_TYPE, DWH_NUM_NODES, DWH_NODE_TYPE, DWH_CLUSTER_IDENTIFI
                     })
```

```
Out[68]:                     Param       Value
        0  DWH_CLUSTER_TYPE        multi-node
        1  DWH_NUM_NODES           4
        2  DWH_NODE_TYPE           dc2.large
        3  DWH_CLUSTER_IDENTIFIER  dwhCluster
        4  DWH_DB                  dwh
        5  DWH_DB_USER             dwhuser
        6  DWH_DB_PASSWORD         Passw0rd
        7  DWH_PORT                5439
        8  DWH_IAM_ROLE_NAME       dwhRole
```

3.1 Create clients for EC2, S3, IAM, and Redshift

```
In [69]: import boto3

         ec2 =

         s3 =

         iam =

         redshift =
```

3.2 Check out the sample data sources on S3

```
In [73]: sampleDbBucket =  s3.Bucket("awssampledbuswest2")

         # TODO: Iterate over bucket objects starting with "ssbgz" and print
```

```
s3.ObjectSummary(bucket_name='awssampledbuswest2', key='ssbgz/')
s3.ObjectSummary(bucket_name='awssampledbuswest2', key='ssbgz/customer0002_part_00.gz')
s3.ObjectSummary(bucket_name='awssampledbuswest2', key='ssbgz/dwdate.tbl.gz')
s3.ObjectSummary(bucket_name='awssampledbuswest2', key='ssbgz/lineorder0000_part_00.gz')
s3.ObjectSummary(bucket_name='awssampledbuswest2', key='ssbgz/lineorder0001_part_00.gz')
s3.ObjectSummary(bucket_name='awssampledbuswest2', key='ssbgz/lineorder0002_part_00.gz')
s3.ObjectSummary(bucket_name='awssampledbuswest2', key='ssbgz/lineorder0003_part_00.gz')
s3.ObjectSummary(bucket_name='awssampledbuswest2', key='ssbgz/lineorder0004_part_00.gz')
s3.ObjectSummary(bucket_name='awssampledbuswest2', key='ssbgz/lineorder0005_part_00.gz')
s3.ObjectSummary(bucket_name='awssampledbuswest2', key='ssbgz/lineorder0006_part_00.gz')
```

```
s3.ObjectSummary(bucket_name='awssampledbuswest2', key='ssbgz/lineorder0007_part_00.gz')
s3.ObjectSummary(bucket_name='awssampledbuswest2', key='ssbgz/part0000_part_00.gz')
s3.ObjectSummary(bucket_name='awssampledbuswest2', key='ssbgz/part0001_part_00.gz')
s3.ObjectSummary(bucket_name='awssampledbuswest2', key='ssbgz/part0002_part_00.gz')
s3.ObjectSummary(bucket_name='awssampledbuswest2', key='ssbgz/part0003_part_00.gz')
s3.ObjectSummary(bucket_name='awssampledbuswest2', key='ssbgz/supplier.tbl_0000_part_00.gz')
s3.ObjectSummary(bucket_name='awssampledbuswest2', key='ssbgz/supplier0001_part_00.gz')
s3.ObjectSummary(bucket_name='awssampledbuswest2', key='ssbgz/supplier0002_part_00.gz')
s3.ObjectSummary(bucket_name='awssampledbuswest2', key='ssbgz/supplier0003_part_00.gz')
```

3.3 STEP 1: IAM ROLE

- Create an IAM Role that makes Redshift able to access S3 bucket (ReadOnly)

```
In [ ]: # TODO: Create the IAM role
        try:
            print('1.1 Creating a new IAM Role')
            dwhRole =


        except Exception as e:
            print(e)
```

```
In [ ]: # TODO: Attach Policy
        print('1.2 Attaching Policy')
```

```
In [82]: # TODO: Get and print the IAM role ARN
         print('1.3 Get the IAM role ARN')
         roleArn =

         print(roleArn)
```

```
1.1 Creating a new IAM Role
An error occurred (EntityAlreadyExists) when calling the CreateRole operation: Role with name dw
1.2 Attaching Policy
1.3 Get the IAM role ARN
arn:aws:iam::988332130976:role/dwhRole
```

3.4 STEP 2: Redshift Cluster

- Create a RedShift Cluster
- For complete arguments to `create_cluster`, see docs

```
In [83]: try:
             response = redshift.create_cluster(
                 # TODO: add parameters for hardware
```

```
            # TODO: add parameters for identifiers & credentials


            # TODO: add parameter for role (to allow s3 access)

        )
    except Exception as e:
        print(e)
```

```
An error occurred (ClusterAlreadyExists) when calling the CreateCluster operation: Cluster alrea
```

3.5 2.1 *Describe* the cluster to see its status

- run this block several times until the cluster status becomes `Available`

```
In [77]: def prettyRedshiftProps(props):
             pd.set_option('display.max_colwidth', -1)
             keysToShow = ["ClusterIdentifier", "NodeType", "ClusterStatus", "MasterUsername", "
             x = [(k, v) for k,v in props.items() if k in keysToShow]
             return pd.DataFrame(data=x, columns=["Key", "Value"])

         myClusterProps = redshift.describe_clusters(ClusterIdentifier=DWH_CLUSTER_IDENTIFIER)['
         prettyRedshiftProps(myClusterProps)
```

```
Out[77]:                  Key  \
         0  ClusterIdentifier
         1  NodeType
         2  ClusterStatus
         3  MasterUsername
         4  DBName
         5  Endpoint
         6  VpcId
         7  NumberOfNodes

                                                                                              Valu
         0  dwhcluster
         1  dc2.large
         2  available
         3  dwhuser
         4  dwh
         5  {'Address': 'dwhcluster.csmamz5zxmle.us-west-2.redshift.amazonaws.com', 'Port': 5439
         6  vpc-54d40a2c
         7  4
```

2.2 Take note of the cluster endpoint and role ARN
DO NOT RUN THIS unless the cluster status becomes "Available"

```
In [78]: DWH_ENDPOINT = myClusterProps['Endpoint']['Address']
         DWH_ROLE_ARN = myClusterProps['IamRoles'][0]['IamRoleArn']
         print("DWH_ENDPOINT :: ", endpoint)
         print("DWH_ROLE_ARN :: ", roleArn)
```

```
DWH_ENDPOINT ::  dwhcluster.csmamz5zxmle.us-west-2.redshift.amazonaws.com
DWH_ROLE_ARN ::  arn:aws:iam::988332130976:role/dwhRole
```

3.6 STEP 3: Open an incoming TCP port to access the cluster ednpoint

```
In [84]: try:
             vpc = ec2.Vpc(id=myClusterProps['VpcId'])
             defaultSg = list(vpc.security_groups.all())[0]
             print(defaultSg)

             defaultSg.authorize_ingress(
                 GroupName= ,  # TODO: fill out
                 CidrIp='',  # TODO: fill out
                 IpProtocol='',  # TODO: fill out
                 FromPort=int(DWH_PORT),
                 ToPort=int(DWH_PORT)
             )
         except Exception as e:
             print(e)
```

```
ec2.SecurityGroup(id='sg-d6161da0')
An error occurred (InvalidPermission.Duplicate) when calling the AuthorizeSecurityGroupIngress o
```

3.7 STEP 4: Make sure you can connect to the clusterConnect to the cluster

```
In [80]: %load_ext sql
```

```
The sql extension is already loaded. To reload it, use:
  %reload_ext sql
```

```
In [81]: conn_string="postgresql://{}:{}@{}:{}/{}".format(DWH_DB_USER, DWH_DB_PASSWORD, DWH_ENDP
         print(conn_string)
         %sql $conn_string
```

```
postgresql://dwhuser:Passw0rd@dwhcluster.csmamz5zxmle.us-west-2.redshift.amazonaws.com:5439/dwh
```

```
Out[81]: 'Connected: dwhuser@dwh'
```

3.8 STEP 5: Clean up your resources

DO NOT RUN THIS UNLESS YOU ARE SURE We will be using these resources in the next exercises

```
In [85]: #### CAREFUL!!
         #-- Uncomment & run to delete the created resources
         redshift.delete_cluster( ClusterIdentifier=DWH_CLUSTER_IDENTIFIER,  SkipFinalClusterSna
         #### CAREFUL!!
```

```
Out[85]: {'Cluster': {'AllowVersionUpgrade': True,
           'AutomatedSnapshotRetentionPeriod': 1,
           'AvailabilityZone': 'us-west-2b',
           'ClusterCreateTime': datetime.datetime(2019, 2, 16, 6, 21, 30, 630000, tzinfo=tzutc()
           'ClusterIdentifier': 'dwhcluster',
           'ClusterParameterGroups': [{'ParameterApplyStatus': 'in-sync',
             'ParameterGroupName': 'default.redshift-1.0'}],
           'ClusterSecurityGroups': [],
           'ClusterStatus': 'deleting',
           'ClusterSubnetGroupName': 'default',
           'ClusterVersion': '1.0',
           'DBName': 'dwh',
           'Encrypted': False,
           'Endpoint': {'Address': 'dwhcluster.csmamz5zxmle.us-west-2.redshift.amazonaws.com',
            'Port': 5439},
           'EnhancedVpcRouting': False,
           'IamRoles': [{'ApplyStatus': 'in-sync',
             'IamRoleArn': 'arn:aws:iam::988332130976:role/dwhRole'}],
           'MasterUsername': 'dwhuser',
           'NodeType': 'dc2.large',
           'NumberOfNodes': 4,
           'PendingModifiedValues': {},
           'PreferredMaintenanceWindow': 'fri:10:30-fri:11:00',
           'PubliclyAccessible': True,
           'Tags': [],
           'VpcId': 'vpc-54d40a2c',
           'VpcSecurityGroups': []},
          'ResponseMetadata': {'HTTPHeaders': {'content-length': '2041',
            'content-type': 'text/xml',
            'date': 'Sat, 16 Feb 2019 07:13:32 GMT',
            'x-amzn-requestid': '5e58b2d8-31ba-11e9-b19b-0945d449b0a9'},
           'HTTPStatusCode': 200,
           'RequestId': '5e58b2d8-31ba-11e9-b19b-0945d449b0a9',
           'RetryAttempts': 0}}
```

- run this block several times until the cluster really deleted

```
In [86]: myClusterProps = redshift.describe_clusters(ClusterIdentifier=DWH_CLUSTER_IDENTIFIER)['
         prettyRedshiftProps(myClusterProps)
```

```
Out[86]:                  Key  \
         0  ClusterIdentifier
         1  NodeType
         2  ClusterStatus
         3  MasterUsername
         4  DBName
         5  Endpoint
         6  VpcId
         7  NumberOfNodes

                                                                                        Valu
         0  dwhcluster
         1  dc2.large
         2  deleting
         3  dwhuser
         4  dwh
         5  {'Address': 'dwhcluster.csmamz5zxmle.us-west-2.redshift.amazonaws.com', 'Port': 5439
         6  vpc-54d40a2c
         7  4
```

```
In [87]: #### CAREFUL!!
         #-- Uncomment & run to delete the created resources
         iam.detach_role_policy(RoleName=DWH_IAM_ROLE_NAME, PolicyArn="arn:aws:iam::aws:policy/A
         iam.delete_role(RoleName=DWH_IAM_ROLE_NAME)
         #### CAREFUL!!
```

```
Out[87]: {'ResponseMetadata': {'HTTPHeaders': {'content-length': '200',
            'content-type': 'text/xml',
            'date': 'Sat, 16 Feb 2019 07:13:50 GMT',
            'x-amzn-requestid': '694f8d91-31ba-11e9-9438-d3ce9c613ef8'},
           'HTTPStatusCode': 200,
           'RequestId': '694f8d91-31ba-11e9-9438-d3ce9c613ef8',
           'RetryAttempts': 0}}
```

```
In [ ]:
```