L3 Exercise 4 - Table Design

August 1, 2021

1 Exercise 4: Optimizing Redshift Table Design

In [26]:
```
%load_ext sql
```

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

In [27]:
```
from time import time
import configparser
import matplotlib.pyplot as plt
import pandas as pd
import json
```

In [28]:
```
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[28]:

	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

In [29]:
```
# Create clients for EC2, S3, IAM, and Redshift
import boto3

ec2 = boto3.resource('ec2',
                       region_name="us-east-1",
                       aws_access_key_id=KEY,
                       aws_secret_access_key=SECRET
                    )

s3 = boto3.resource('s3',
                       region_name="us-east-1",
                       aws_access_key_id=KEY,
                       aws_secret_access_key=SECRET
                    )

iam = boto3.client('iam',aws_access_key_id=KEY,
                     aws_secret_access_key=SECRET,
                     region_name='us-east-1'
                  )

redshift = boto3.client('redshift',
                       region_name="us-east-1",
                       aws_access_key_id=KEY,
                       aws_secret_access_key=SECRET
                       )
```

In [30]:
```
sampleDbBucket =  s3.Bucket("awssampledbuswest2")

# TODO: Iterate over bucket objects starting with "ssbgz" and print
for obj in sampleDbBucket.objects.filter(Prefix="ssbgz"):
    print(obj)
```

```
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')
```

In [31]:
```
# IAM ROLE
# Create an IAM Role that makes Redshift able to access S3 bucket (ReadOnly)
from botocore.exceptions import ClientError

#1.1 Create the role,
try:
    print("1.1 Creating a new IAM Role")
    dwhRole = iam.create_role(
        Path='/',
        RoleName=DWH_IAM_ROLE_NAME,
        Description = "Allows Redshift clusters to call AWS services on your behalf.",
        AssumeRolePolicyDocument=json.dumps(
            {'Statement': [{'Action': 'sts:AssumeRole',
               'Effect': 'Allow',
               'Principal': {'Service': 'redshift.amazonaws.com'}}],
             'Version': '2012-10-17'})
    )
except Exception as e:
    print(e)


print("1.2 Attaching Policy")

iam.attach_role_policy(RoleName=DWH_IAM_ROLE_NAME,
                       PolicyArn="arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess"
                      )['ResponseMetadata']['HTTPStatusCode']

print("1.3 Get the IAM role ARN")
roleArn = iam.get_role(RoleName=DWH_IAM_ROLE_NAME)['Role']['Arn']
```

```
        print(roleArn)
```

```
1.1 Creating a new IAM Role
1.2 Attaching Policy
1.3 Get the IAM role ARN
arn:aws:iam::643266055682:role/dwhRole
```

```
In [32]: # TODO: Attach Policy
         print('1.2 Attaching Policy')
         iam.attach_role_policy(RoleName=DWH_IAM_ROLE_NAME,
                                PolicyArn="arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess"
                               )['ResponseMetadata']['HTTPStatusCode']

         print("1.3 Get the IAM role ARN")
         roleArn = iam.get_role(RoleName=DWH_IAM_ROLE_NAME)['Role']['Arn']

         print(roleArn)
```

```
1.2 Attaching Policy
1.3 Get the IAM role ARN
arn:aws:iam::643266055682:role/dwhRole
```

```
In [33]: # TODO: Get and print the IAM role ARN
         print('1.3 Get the IAM role ARN')
         roleArn = iam.get_role(RoleName=DWH_IAM_ROLE_NAME)['Role']['Arn']
         print(roleArn)
```

```
1.3 Get the IAM role ARN
arn:aws:iam::643266055682:role/dwhRole
```

```
In [34]: ## create redshift cluster
         try:
             response = redshift.create_cluster(
                 #HW
                 ClusterType=DWH_CLUSTER_TYPE,
                 NodeType=DWH_NODE_TYPE,
                 NumberOfNodes=int(DWH_NUM_NODES),

                 #Identifiers & Credentials
                 DBName=DWH_DB,
                 ClusterIdentifier=DWH_CLUSTER_IDENTIFIER,
                 MasterUsername=DWH_DB_USER,
                 MasterUserPassword=DWH_DB_PASSWORD,

                 #Roles (for s3 access)
```

```
            IamRoles=[roleArn]
        )
    except Exception as e:
        print(e)
```

In [43]:
```
# Describe the cluster to see its status
# run this block several times until the cluster status becomes Available

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[43]:

	Key	Valu
0	ClusterIdentifier	
1	NodeType	
2	ClusterStatus	
3	MasterUsername	
4	DBName	
5	Endpoint	
6	VpcId	
7	NumberOfNodes	
0	dwhcluster	
1	dc2.large	
2	available	
3	dwhuser	
4	dwh	
5	{'Address': 'dwhcluster.cg7gvfu1dyuj.us-east-1.redshift.amazonaws.com', 'Port': 5439	
6	vpc-f3ac448e	
7	4	

In [44]:
```
# Take note of the cluster endpoint and role ARN
DWH_ENDPOINT = myClusterProps['Endpoint']['Address']
DWH_ROLE_ARN = myClusterProps['IamRoles'][0]['IamRoleArn']
print("DWH_ENDPOINT :: ", DWH_ENDPOINT)
print("DWH_ROLE_ARN :: ", DWH_ROLE_ARN)

# myClusterProps
```

```
DWH_ENDPOINT ::  dwhcluster.cg7gvfu1dyuj.us-east-1.redshift.amazonaws.com
DWH_ROLE_ARN ::  arn:aws:iam::643266055682:role/dwhRole
```