In [ ]:
```
# Importing libraries:
import pandas as pd
import boto3
import psycopg2
import configparser
```

In [ ]:
```
# Reading configuration file:
config = configparser.ConfigParser()
config.read_file(open('cluster.config'))
```

In [ ]:
```
# Retrieving AWS credentials from the configuration file:
KEY                    = config.get('AWS','KEY')
SECRET                 = config.get('AWS','SECRET')

# Retrieving Redshift cluster configuration from the configuration file:
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')
```

In [ ]:
```
# Creating a DataFrame to store the Redshift cluster configuration parameters:
pd.DataFrame({"Param":["DWH_CLUSTER_TYPE","DWH_NUM_NODES","DWH_NODE_TYPE","DWH_CLUSTER_IDENTIFIER","DWH_DB","DWH_DB_USER","DWH_DB_PASSWORD","DWH_PORT"
,"DWH_IAM_ROLE_NAME"],
              "Value":[DWH_CLUSTER_TYPE,DWH_NUM_NODES,DWH_NODE_TYPE,DWH_CLUSTER_IDENTIFIER,DWH_DB,DWH_DB_USER,DWH_DB_PASSWORD,DWH_PORT,DWH_IAM_ROLE_NAME]})
```

In [ ]:
```
# Creating AWS clients for Redshift and IAM:
redshift = boto3.client('redshift',
                    region_name='eu-west-2',
                    aws_access_key_id=KEY,
                    aws_secret_access_key=SECRET)

iam = boto3.client('iam',
                    region_name='eu-west-2',
                    aws_access_key_id=KEY,
                    aws_secret_access_key=SECRET)
```

In [ ]:
```
# Retrieving the IAM role ARN:
roleArn = iam.get_role(RoleName=DWH_IAM_ROLE_NAME)['Role']['Arn']
```

In [ ]:
```
# Creating a Redshift cluster:
try:
    response = redshift.create_cluster(
        ClusterType = DWH_CLUSTER_TYPE,
        NodeType = DWH_NODE_TYPE,

        #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 [ ]:
```
# Retrieving information about the created Redshift cluster:
redshift.describe_clusters(ClusterIdentifier=DWH_CLUSTER_IDENTIFIER)['Clusters'][0]
```

In [ ]:
```
# Defining a function to display selected Redshift cluster properties:
def prettyRedshiftProps(props):
        pd.set_option('display.max_colwidth',-1)
        keysToShow = ["ClusterIdentifier","NodeType","ClusterStatus","MasterUsername","DBName","Endpoint","VpcId"]
        x=[(k,v) for k,v in props.items() if k in keysToShow]
        return pd.DataFrame(data=x, columns=["Key","Value"])

# Calling the prettyRedshiftProps function to display the selected Redshift cluster properties:
myClusterProps = redshift.describe_clusters(ClusterIdentifier=DWH_CLUSTER_IDENTIFIER)['Clusters'][0]
prettyRedshiftProps(myClusterProps)
```

In [ ]:
```
# Extracting relevant information from the Redshift cluster properties:
DWH_ENDPOINT = myClusterProps['Endpoint']['Address']
DWH_ROLE_ARN = myClusterProps['IamRoles'][0]['IamRoleArn']
DB_NAME = myClusterProps['DBName']
DB_USER = myClusterProps['MasterUsername']
```