Documetation

List of regions: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-regions-availability-zones.html#concepts-regions

Code to find aws regions (opt in not required): (17/28)regions

///////////////////////////////////////////////////////////////////////////////////////////////////

import boto3  
import os  
  
# Set AWS credentials as environment variables  
os.environ['AWS\_ACCESS\_KEY\_ID'] = 'YOUR\_ACCESS\_KEY'  
os.environ['AWS\_SECRET\_ACCESS\_KEY'] = 'YOUR\_SECRET\_KEY'  
  
# Initialize the EC2 client with a specific region  
ec2\_client = boto3.client('ec2', region\_name='us-east-1') # Replace with your desired region  
  
# Get the list of all AWS regions  
response = ec2\_client.describe\_regions()  
  
# Extract region names from the response  
region\_names = [region['RegionName'] for region in response['Regions']]  
  
# Print the list of region names  
for region\_name in region\_names:  
 print(region\_name)

///////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

Code to find aws regions (opt in not required and required): (28/28)regions

import boto3  
import os  
  
# Set AWS credentials as environment variables  
os.environ['AWS\_ACCESS\_KEY\_ID'] = 'AKIAZKZXRMJ3RH5SUFH7'  
os.environ['AWS\_SECRET\_ACCESS\_KEY'] = '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'  
  
# Initialize the EC2 client with a specific region  
ec2\_client = boto3.client('ec2', region\_name='ca-central-1') # Replace with your desired region  
  
# Get the list of all AWS regions  
response = boto3.session.Session().get\_available\_regions('ec2')  
count=0  
l1=[]  
for region\_name in response:  
 count+=1  
 l1.append(region\_name)  
for each in l1:  
 print(each)  
  
print(count)

code to get 28 regions and services under each region:

import boto3  
import os  
  
  
def list\_regions\_and\_services():  
 os.environ['AWS\_ACCESS\_KEY\_ID'] = 'AKIAZKZXRMJ3RH5SUFH7'  
 os.environ['AWS\_SECRET\_ACCESS\_KEY'] = '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'  
  
  
 # Initialize the Boto3 client for AWS service "ec2" in the us-east-1 region.  
 ec2\_client = boto3.client('ec2', region\_name='us-east-1')  
  
 # Get a list of all AWS regions using the ec2\_client.  
 regions = boto3.session.Session().get\_available\_regions('ec2')  
  
 # Iterate through each region and find the services available.  
 p=0 #counting total iterations or results produced  
 c=0 #number of regions  
 for region in regions:  
 print(f"Region: {region}")  
 c+=1  
  
 # Initialize a Boto3 client for each service in the current region.  
 session = boto3.Session(region\_name=region)  
 available\_services = session.get\_available\_services()  
 k=0 #number of services under each region  
 # Print the list of services available in the current region.  
 for service in available\_services:  
 print(f"- {service}")  
 k+=1  
 p+=1  
  
 print("id region=",c)  
 print("no.of services=",k)  
 print("/////////////////////////////////////////////////////////////////////")  
 print("total results produced=",p)  
if \_\_name\_\_ == "\_\_main\_\_":  
 list\_regions\_and\_services()

Q. difference btw boto3.resource and client:

Regions and services (short): <https://www.saisci.com/aws/listing-aws-regions-using-boto3-python/#OptInStatus>

overall code:

import boto3  
import json  
import pprint  
from pymongo import MongoClient  
  
# Initialize the AWS Pricing client for the 'us-east-1' region  
pricing = boto3.client('pricing', region\_name='us-east-1')  
  
all\_services = [] #has service code and its attributes  
next\_token = None  
  
while True:  
 params = {}  
 if next\_token:  
 params['NextToken'] = next\_token  
  
 response = pricing.describe\_services(\*\*params)  
 all\_services.extend(response['Services'])  
 next\_token = response.get('NextToken')  
 if not next\_token:  
 break  
  
  
serviceCodeList=[]  
for service in all\_services:  
 serviceCodeList.append(service['ServiceCode']) #we are only adding service code and not attributes  
  
print(serviceCodeList)  
f=0  
c=0  
for each in serviceCodeList:  
 c+=1  
 print(c,"",each," Products")  
 #print("=====================")  
  
 #c = 0 # Counter to track the number of products  
 next\_token = "" # Initialize the NextToken as an empty string  
  
 while True:  
 # Use the get\_products method to retrieve pricing information for the 'nimble' service  
 response = pricing.get\_products(  
 ServiceCode=each,  
 Filters=[  
 {'Type': 'TERM\_MATCH', 'Field': 'serviceCode', 'Value': each},  
 ],  
 NextToken=next\_token # Pass the NextToken to fetch the next page of results  
 )  
 #add to mongoDB  
 # MongoDB connection string (replace with your own)  
 connection\_string = "mongodb://localhost:27017"  
  
 # Database and collection names (replace with your own)  
 db\_name = "AWS"  
 collection\_name = 'aws\_pricing'  
  
 try:  
 # Connect to MongoDB  
 client = MongoClient(connection\_string)  
  
 # Access the database  
 db = client[db\_name]  
  
 # Access the collection  
 collection = db[collection\_name]  
 # Insert the JSON data into the collection  
 for price in response['PriceList']:  
 pp = json.loads(price)  
 inserted\_data = collection.insert\_one(pp)  
 f += 1  
 #print(f)  
  
  
 except Exception as e:  
 print("Error:", e)  
 finally:  
 client.close()  
 '''  
 # Loop through the list of pricing information  
 for price in response['PriceList']:  
 pp = pprint.PrettyPrinter(indent=1, width=300)  
 pp.pprint(json.loads(price))  
 c += 1  
 f+=1  
 print(f)  
 print()  
'''  
 # Check if there are more pages of results  
 if 'NextToken' in response:  
 next\_token = response['NextToken']  
 else:  
 break  
  
 c += 1

the above code initially gets all the services and stores them in a list names “all\_services” later an api call is made to call all the pricing details of each service in the “all\_services” list

these details are fetched through api calls from boto3 and are stored in mongodb file.

Km