**AWS Cloud Billing Reports:**

**AWS CLI Command to download billing reports for a particular month –** “aws ce get-cost-and-usage --time-period Start=2023-09-01,End=2023-09-30 --granularity MONTHLY --metrics BlendedCost --group-by Type=DIMENSION,Key=SERVICE --output json > report.json”

We can download billing reports in four formats

* Text
* Json
* Yaml
* Table

**This shell script will download all the billing reports for the calender year**

for MONTH in {01..12}; do

YEAR=$(date +"%Y")

LAST\_DAY=$(cal $MONTH $YEAR | awk 'NF {DAYS = $NF} END {print DAYS}')

START\_DATE="${YEAR}-${MONTH}-01"

END\_DATE="${YEAR}-${MONTH}-${LAST\_DAY}"

# Check if this is the last month to handle December

if [ "$MONTH" -eq 12 ]; then

END\_DATE="${YEAR}-${MONTH}-31"

fi

echo "Generating report for ${START\_DATE} to ${END\_DATE}..."

# Execute the AWS CLI command for the current month

aws ce get-cost-and-usage \

--time-period "Start=${START\_DATE},End=${END\_DATE}" \

--granularity MONTHLY \

--metrics BlendedCost \

--group-by Type=DIMENSION,Key=SERVICE \

--output text \

> report\_${YEAR}\_${MONTH}.csv

Done

**This python will convert the json file billing report to csv file:**

import pandas as pd

import json

# Load the JSON data from the file

with open('report.json', 'r') as f:

data = json.load(f)

# Extract relevant information

groups = data['ResultsByTime'][0]['Groups']

keys = [group['Keys'][0] for group in groups]

amounts = [group['Metrics']['BlendedCost']['Amount'] for group in groups]

units = [group['Metrics']['BlendedCost']['Unit'] for group in groups]

# Create a DataFrame

df = pd.DataFrame({'Keys': keys, 'Amount': amounts, 'Unit': units})

# Save to CSV

df.to\_csv('report.csv', index=False)

**Once the Billing reports are converted to csv format we can Use PowerBI to generate graphs but we need to give the input manually and to do it automatically we need a license.**

**Azure DevOps yaml pipeline script to download the billing reports and we can store them in S3 buckets using the following script:**

trigger:

- main

pool:

  vmImage: ubuntu-latest

steps:

- task: UsePythonVersion@0

  displayName: 'Use Python 3.x'

  inputs:

    versionSpec: '3.x'

    addToPath: true

- script: |

    python -m pip install --upgrade pip

    pip install pandas

    aws configure set aws\_access\_key\_id $AWS\_ACCESS\_KEY\_ID

    aws configure set aws\_secret\_access\_key $AWS\_SECRET\_ACCESS\_KEY

    aws configure set region us-east-1

  displayName: 'Configure AWS CLI'

  env:

    AWS\_ACCESS\_KEY\_ID: $(AWS\_ACCESS\_KEY\_ID)

    AWS\_SECRET\_ACCESS\_KEY: $(AWS\_SECRET\_ACCESS\_KEY)

- script: |

    aws ce get-cost-and-usage --time-period Start=2023-09-01,End=2023-09-30 --granularity MONTHLY --metrics BlendedCost --group-by Type=DIMENSION,Key=SERVICE --output json > report.json

  displayName: 'Get Cost and Usage'

- task: PublishBuildArtifacts@1

  inputs:

    pathToPublish: 'report.json'

    artifactName: 'cost-report'

- task: PythonScript@0

  inputs:

    scriptSource: 'inline'

    script: |

      import pandas as pd

      import json

      # Load the JSON data from the file

      with open('report.json', 'r') as f:

          data = json.load(f)

      # Extract relevant information

      groups = data['ResultsByTime'][0]['Groups']

      keys = [group['Keys'][0] for group in groups]

      amounts = [group['Metrics']['BlendedCost']['Amount'] for group in groups]

      units = [group['Metrics']['BlendedCost']['Unit'] for group in groups]

      # Create a DataFrame

      df = pd.DataFrame({'Keys': keys, 'Amount': amounts, 'Unit': units})

      # Save to CSV

      df.to\_csv('report.csv', index=False)

- task: PublishBuildArtifacts@1

  inputs:

    pathToPublish: 'report.csv'

    artifactName: 'cost-report-csv'

- task: AmazonWebServices.aws-vsts-tools.S3Upload.S3Upload@1

  inputs:

    awsCredentials: 'AWS\_SC'

    regionName: 'us-east-1'

    bucketName: 'awsbillingsreport112245'

    sourceFolder: '/home/vsts/work/1/s'

    globExpressions: '\*\*report.csv'

    createBucket: true

    keyManagement: 'awsManaged'

    encryptionAlgorithm: 'AES256'