Steps involved in execution of Bash script on Ubuntu machine:

1. Created an Ubuntu machine in Azure Portal
2. Install JQ on Ubuntu using the command “snap install jq”
3. Install azure cli using the commands

* sudo apt-get update
* sudo apt-get install azure-cli

1. Create a directory on ubuntu machine using mkdir “project”
2. Create a file inside the directory “project” with the filename “script.sh”
3. Copy and paste the script inside the file “script.sh”

Script:

#! /usr/bin/bash

set -o errexit

set -o nounset

set -o pipefail

export ARM\_SUBSCRIPTION\_ID= XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX

export ARM\_TENANT\_ID= XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX

export ARM\_CLIENT\_ID=XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX

export ARM\_CLIENT\_SECRET=XXXXXXXXXXXXX

export MANAGEMENT\_RESOURCE\_ENDPOINT="https://management.core.windows.net/" # This is Fixed value (DO NOT CHANGE)

export AZURE\_DATABRICKS\_APP\_ID="2ff814a6-3304-4ab8-85cb-cd0e6f879c1d" # This is Fixed value (DO NOT CHANGE)

export RESOURCE\_GROUP="rg-test-01"

export LOCATION="southeastasia"

export DATABRICKS\_WORKSPACE="TestWorkspace"

export DATABRICKS\_CLUSTER\_NAME="test-cluster-01"

export DATABRICKS\_SPARK\_VERSION="7.3.x-scala2.12"

export DATABRICKS\_NODE\_TYPE="Standard\_D3\_v2"

export DATABRICKS\_NUM\_WORKERS=3 # Need to be number

export DATABRICKS\_SPARK\_CONF='{"spark.speculation":"true","spark.databricks.delta.preview.enabled":"true"}' # Needs to be valid JSON

export DATABRICKS\_AUTO\_TERMINATE\_MINUTES=60 # Need to be a number

# Login using service principle

echo "Logging in using Azure service priciple"

az login --service-principal -u $ARM\_CLIENT\_ID -p $ARM\_CLIENT\_SECRET --tenant $ARM\_TENANT\_ID

az account set -s  $ARM\_SUBSCRIPTION\_ID

# Create Resource Group if not exists

# NOTE: you can get list of az location from "az account list-locations | jq .[].name"

if [[ $(az group exists --resource-group $RESOURCE\_GROUP) = "false" ]]; then

    echo "Resource Group does not exists, so creating.."

    az group create --name $RESOURCE\_GROUP --location $LOCATION

fi

# Enable install of extensions without prompt

az config set extension.use\_dynamic\_install=yes\_without\_prompt

# Create databricks workspace using extenstion

# The extension will automatically install the first time you run an az databricks workspace command

# Ref: https://docs.microsoft.com/en-us/cli/azure/ext/databricks/databricks?view=azure-cli-latest

if [[ $(az databricks workspace list | jq .[].name | grep -w $DATABRICKS\_WORKSPACE) = $DATABRICKS\_WORKSPACE ]]; then

    echo "Databricks workspace does not exists, so creating.."

    az databricks workspace create \

        --location $LOCATION \

        --name $DATABRICKS\_WORKSPACE \

        --sku trial \

        --resource-group $RESOURCE\_GROUP \

        --enable-no-public-ip \

        --tags environment=demo level=level3

fi

# Get workspace id in the given resource group e.g. /subscriptions/(subscription\_id)/resourceGroups/(rg)/providers/Microsoft.Databricks/workspaces/(databricks\_workspace)

wsId=$(az resource show --resource-type Microsoft.Databricks/workspaces -g $RESOURCE\_GROUP -n "$DATABRICKS\_WORKSPACE" --query id -o tsv)

echo "Workspce ID: $wsId"

# Get workspace url e.g. adb-xxxxxxxxxxxxxxxx.x.azuredatabricks.net

workspaceUrl=$(az resource show --resource-type Microsoft.Databricks/workspaces -g "$RESOURCE\_GROUP" -n "$DATABRICKS\_WORKSPACE" --query properties.workspaceUrl --output tsv)

echo "Workspce URL: $workspaceUrl"

# token response for the azure databricks app

token\_response=$(az account get-access-token --resource $AZURE\_DATABRICKS\_APP\_ID)

echo $token\_response

# Extract accessToken value

token=$(jq .accessToken -r <<< "$token\_response")

echo "Token: $token"

# Get the Azure Management Resource endpoint token

# https://docs.microsoft.com/en-us/azure/databricks/dev-tools/api/latest/aad/service-prin-aad-token#--get-the-azure-management-resource-endpoint-token

az\_mgmt\_resource\_endpoint=$(curl -X GET -H 'Content-Type: application/x-www-form-urlencoded' \

-d 'grant\_type=client\_credentials&client\_id='$ARM\_CLIENT\_ID'&resource='$MANAGEMENT\_RESOURCE\_ENDPOINT'&client\_secret='$ARM\_CLIENT\_SECRET \

https://login.microsoftonline.com/$ARM\_TENANT\_ID/oauth2/token)

# Extract the access\_token value

mgmt\_access\_token=$(jq .access\_token -r <<< "$az\_mgmt\_resource\_endpoint" )

echo "Management Access Token: $mgmt\_access\_token"

# Create PAT token valid for 5 min (300 sec)

pat\_token\_response=$(curl -X POST \

    -H "Authorization: Bearer $token" \

    -H "X-Databricks-Azure-SP-Management-Token: $mgmt\_access\_token" \

    -H "X-Databricks-Azure-Workspace-Resource-Id: $wsId" \

    -d '{"lifetime\_seconds": 300,"comment": "this is an example token"}' \

    https://$workspaceUrl/api/2.0/token/create

)

# Print PAT token

pat\_token=$(jq .token\_value -r <<< "$pat\_token\_response")

echo $pat\_token

# List PAT tokens (OPTIONAL)

curl -X GET \

    -H "Authorization: Bearer $token" \

    -H "X-Databricks-Azure-SP-Management-Token: $mgmt\_access\_token" \

    -H "X-Databricks-Azure-Workspace-Resource-Id: $wsId" \

    https://$workspaceUrl/api/2.0/token/list

# List current clusters (OPTIONAL) and could be used to determine the next command e.g. create,restart,terminate etc

curl -X GET \

    -H "Authorization: Bearer $token" \

    -H "X-Databricks-Azure-SP-Management-Token: $mgmt\_access\_token" \

    -H "X-Databricks-Azure-Workspace-Resource-Id: $wsId" \

    https://$workspaceUrl/api/2.0/clusters/list

# Create Cluster config from variables

JSON\_STRING=$( jq -n -c \

                --arg cn "$DATABRICKS\_CLUSTER\_NAME" \

                --arg sv "$DATABRICKS\_SPARK\_VERSION" \

                --arg nt "$DATABRICKS\_NODE\_TYPE" \

                --arg nw "$DATABRICKS\_NUM\_WORKERS" \

                --arg sc "$DATABRICKS\_SPARK\_CONF" \

                --arg at "$DATABRICKS\_AUTO\_TERMINATE\_MINUTES" \

                '{cluster\_name: $cn,

                spark\_version: $sv,

                node\_type\_id: $nt,

                num\_workers: ($nw|tonumber),

                autotermination\_minutes: ($at|tonumber),

                spark\_conf: ($sc|fromjson)}' )

# Create a new Cluster

# Reference: https://docs.microsoft.com/en-us/azure/databricks/dev-tools/api/latest/

cluster\_id\_response=$(curl -X POST \

    -H "Authorization: Bearer $token" \

    -H "X-Databricks-Azure-SP-Management-Token: $mgmt\_access\_token" \

    -H "X-Databricks-Azure-Workspace-Resource-Id: $wsId" \

    -d $JSON\_STRING \

    https://$workspaceUrl/api/2.0/clusters/create)

# Print cluster\_id

cluster\_id=$(jq .cluster\_id -r <<< "$cluster\_id\_response")

echo "Cluster id: $cluster\_id"

1. Execute the script using the command “bash script.sh”
2. Fill in these details correctly before executing the script

export ARM\_SUBSCRIPTION\_ID= XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX

export ARM\_TENANT\_ID= XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX

export ARM\_CLIENT\_ID=XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX

export ARM\_CLIENT\_SECRET=XXXXXXXXXXXXX