Working with Azure CLI

Practical Tutorial using Azure Resource Manager & Docker Microservices
J. Imrich - Azure Lab5 11/09/2017
>>>

Download Azure CLI to my local ubuntu Bash shell on Windows 10.

logged into CLI: ‘az login’

PS C:\Users\mcazure> az

```
     /\
    /  \    _____   _ _  ___ _
   / /\ \  |_  / | | | \'__/ _ \
  / ____ \  / /| |_| | | |  __/
 /_/    \_\/___|\__,_|_|  \___|


Welcome to the cool new Azure CLI!
Here are the base commands:

    account          : Manage Azure subscription information.
    acr              : Manage Azure Container Registries.
    acs              : Manage Azure Container Services.
    ad               : Synchronize on-premises directories and manage Azure Active Directory
                       resources.
    aks              : Manage Kubernetes clusters.
    appservice       : Manage App Service plans.
    backup           : Commands to manage Azure Backups.
    batch            : Manage Azure Batch.
    batchai          : Batch AI.
    billing          : Manage Azure Billing.
    cdn              : Manage Azure Content Delivery Networks (CDNs).
    cloud            : Manage registered Azure clouds.
    cognitiveservices: Manage Azure Cognitive Services accounts.
    component        : Manage and update Azure CLI 2.0 components.
    configure        : Display and manage the Azure CLI 2.0 configuration. This command is
                       interactive.
    consumption      : Manage consumption of Azure resources.
    container        : (PREVIEW) Manage Azure Container Instances.
    cosmosdb         : Manage Azure Cosmos DB database accounts.
    disk             : Manage Azure Managed Disks.
    dla              : (PREVIEW) Manage Data Lake Analytics accounts, jobs, and catalogs.
    dls              : (PREVIEW) Manage Data Lake Store accounts and filesystems.
    eventgrid        : Manage Azure Event Grid topics and subscriptions.
    extension        : Manage and update CLI extensions.
    feature          : Manage resource provider features.
    feedback         : Loving or hating the CLI?  Let us know!
    find             : Find Azure CLI commands.
    functionapp      : Manage function apps.
    group            : Manage resource groups and template deployments.
    image            : Manage custom virtual machine images.
    interactive      : Start interactive mode.
    iot              : (PREVIEW) Manage Internet of Things (IoT) assets.
    keyvault         : Safeguard and maintain control of keys, secrets, and certificates.
    lab              : Manage Azure DevTest Labs.
    lock             : Manage Azure locks.
    login            : Log in to Azure.
    logout           : Log out to remove access to Azure subscriptions.
    managedapp       : Manage template solutions provided and maintained by Independent Software
                       Vendors (ISVs).
    monitor          : Manage the Azure Monitor Service.
    mysql            : Manage Azure Database for MySQL servers.
    network          : Manage Azure Network resources.
    policy           : Manage resource policies.
    postgres         : Manage Azure Database for PostgreSQL servers.
    provider         : Manage resource providers.
    redis            : Access to a secure, dedicated Redis cache for your Azure applications.
```

```
    resource         : Manage Azure resources.
    role             : Manage user roles for access control with Azure Active Directory and service
                       principals.
    sf               : Manage and administer Azure Service Fabric clusters.
    snapshot         : Manage point-in-time copies of managed disks, native blobs, or other
                       snapshots.
    sql              : Manage Azure SQL Databases and Data Warehouses.
    storage          : Manage Azure Cloud Storage resources.
    tag              : Manage resource tags.
    vm               : Provision Linux or Windows virtual machines.
    vmss             : Manage groupings of virtual machines in an Azure Virtual Machine Scale Set
                       (VMSS).
    webapp           : Manage web apps.
```

PS C:\Users\mcazure> az login

```
To sign in, use a web browser to open the page https://aka.ms/devicelogin and enter the code XXXXX to
authenticate (sent from your email address)
[
  {
    "cloudName": "AzureCloud",
    "id": "xxxxxx-xxxx",
    "isDefault": true,
    "name": "Free Trial",
    "state": "Enabled",
    "tenantId": "xxxx-xxxx",
    "user": {
      "name": "McKessonUser@email.com",
      "type": "user"
    }
  }
]
```

PS C:\Users\mcazure>

az group create -g mckarm -l eastus
az sql server create -h
az sql server create -n mckdasqlsvr -u sqladmin -p mckdasqlsvrMcKeS@N -g mckarm -l eastus
az sql db create -n mckdb -s mckdasqlsvr -g mckarm

creation of a firewall rule to allow access service broadband IP
Find your ip address by using via commands for client 111.2.345.6
$ipconfig
$ curl -4 icanhazip.com
$ ip addr show eth0 | grep inet | awk '{ print $2; }' | sed 's/\/.*$//'
az sql server firewall-rule create -n sqlServerAccess --start-ip-address 111.2.345.6--end-ip-address 111.2.345.6 -s mckdasqlsvr -g mckarm

Open port 1433 by creating Windows firewall inbound rule, named sqlserver 1433
which is configure for protocol=TCP, port=1433, action=Allow

Connect to mckdasqlsvr.database.windows.net via SQL Server Management Studio and display mckdb database: DDL/DML insert data & query a couple of rows, select * from table products in mckdb database:

az vm image list -f CentOS --all
az vm image list -l eastus –all
`az vm image list-publishers -l eastus --query "[?starts_with(name, 'Microsoft')]"`

az vm create -n mckwinimage -image Win2016DataCenter -g mckarm

az image create -g mcarm -n mckwinimage --os-type Linux --source /subscriptions/xxxxxxxxxxxxx/resourceGroups/rg1/providers/Microsoft.Compute/snapshots/s1 --data-snapshot /subscriptions/xxxxxxxxxxxxx/resourceGroups/rg/providers/Microsoft.Compute/snapshots/s2


```
{
   "id":
"/Subscriptions/xxxxxxxx/Providers/Microsoft.Compute/Locations/eastus/Publishers/Microsoft.Windows.RemoteDesktop
",
   "location": "eastus",
   "name": "Microsoft.Windows.RemoteDesktop",
   "tags": null
 },
```


PS:/mnt/c/Users/mcazure/Downloads/DA/A4$az vm create -h

Connect to the newly created Win2012 VM via RDC:
Display remote connection to the VM at IP=111.2.345.6
Check / Cleanup Resources

az group exists -g mckarm
az group delete -g mckarm
az provider list --query "[?resourceTypes[?resourceType=='databases']].namespace" --out table

```
Result
--------------------
RavenHq.Db
SuccessBricks.ClearDB
```

az provider show --namespace RavenHq.Db --query "resourceTypes[].resourceType"

```
[
  "databases",
  "operations",
  "listCommunicationPreference",
  "updateCommunicationPreference"
]
```

az provider show --namespace SuccessBricks.ClearDB --query "resourceTypes[?resourceType=='databases'].locations"
az provider register --namespace SuccessBricks.ClearDB
az provider show -n SuccessBricks.ClearDB
az group create -g mckarm -l eastus


```
{
  "id": "/subscriptions/xxxxxxxxxx/resourceGroups/mckarm",
  "location": "eastus",
  "managedBy": null,
  "name": "mckarm",
  "properties": {
    "provisioningState": "Succeeded"
  },
  "tags": null
}
```

Working with Azure Resource Manager Templates

Created a template, mydeploy.json

```
{ "$schema": "http://schema.management.azure.com/schemas/2014-04-01-preview/deploymentTemplate.json#",
 "contentVersion": "1.0.0.0",
 "parameters": {},
 "resources": [
 {  "apiVersion": "2014-04-01",
         "name": "mckdb",
        "location": "East US",
         "type": "SuccessBricks.ClearDB/databases", } ] }
```

az group deployment create --resource-group mckarm --template-file mydeploy.json --verbose

```
{ "$schema": "http://schema.management.azure.com/schemas/2014-04-01-preview/deploymentTemplate.json#",
 "contentVersion": "1.0.0.0",  "parameters": {},
 "resources": [ { "apiVersion": "2014-04-01", "name": "mckdb",  "location": "East US",  "type":
"SuccessBricks.ClearDB/databases",  "plan":
                {"name": "Free",  "product": "databases",  "publisher": "cleardb"} } ] }
```

Test Deploy using Parameters

az group deployment create --resource-group mckarm --template-file mydeploy2.json --verbose
Deploy template 'mydeploy2.json'
when prompted for 'resourceName' parameter: … enter resource name

```
{ "$schema": "http://schema.management.azure.com/schemas/2014-04-01-preview/deploymentTemplate.json#",
"contentVersion": "1.0.0.0",
"parameters": { "resourceName": { "type": "string" }},
"resources":
            [ { "apiVersion": "2014-04-01",
             "name":
            "[parameters('resourceName')]",
             "location": "East US",
            "type": "SuccessBricks.ClearDB/databases",
             "plan": { "name": "Free", "product": "databases",
            "publisher": "cleardb" } }
            ] }
```

$ az group deployment list -g mckarm
$ az group delete -g mckarm

Working with VMWare **Follow directions in Week 4 Course Materials "Install+Docker+CE+on+CentOS7.4.docx"**

Download and start VMWare Workstation 12, 30 day trial.
Started CentOS7.4 VM with 2 Processors and 3.2 GB of memory.
Open terminal
sudo yum install git
sudo yum install -y yum-utils device-mapper-persistent-data lvm2
sudo yum-config-manager -add-repo https://download.docker.com/linux/centos/docker-ce.repo
sudo yum install docker-ce
sudo docker search centos
sudo docker pull openshift/base-centos7
sudo docker run -i -t openshift/base-centos7 /bin/bash
sudo docker images
sudo docker ps
sudo ls -l /var/lib/docker