Common Azure CLI Commands

Tuesday, October 26, 2021 8:50 AM

==

Deploy a Spring Boot app to Azure

```
# show Azure account
az account show

# create resource group
az group create \
   --name $AZ_RESOURCE_GROUP \
   --location $AZ_LOCATION \
   | jq

# create a small MySQL server with 1 CPU & 2 GB of RAM
az mysql server create \
   --resource-group $AZ_RESOURCE_GROUP \
   --name $AZ_DATABASE_NAME \
   --location $AZ_LOCATION \
   --sku-name B_Gen5_1 \
   --storage-size 5120 \
   --admin-user $AZ_MYSQL_USERNAME \
   --admin-password $AZ_MYSQL_PASSWORD \
   | jq

# create firewall rule to allow MySQL server access from local IP address
az mysql server firewall-rule create \
   --resource-group $AZ_RESOURCE_GROUP \
   --name $AZ_DATABASE_NAME-database-allow-local-ip \
   --server-name $AZ_DATABASE_NAME \
   --start-ip-address $AZ_LOCAL_IP_ADDRESS \
   --end-ip-address $AZ_LOCAL_IP_ADDRESS \
   | jq

# create firewall rule to allow MySQL server access from Azure resources
az mysql server firewall-rule create \
   --resource-group $AZ_RESOURCE_GROUP \
   --name allAzureIPs \
   --server-name $AZ_DATABASE_NAME \
   --start-ip-address 0.0.0.0 --end-ip-address 0.0.0.0 \
   | jq

# create new demo database
az mysql db create \
   --resource-group $AZ_RESOURCE_GROUP \
   --name demo \
   --server-name $AZ_DATABASE_NAME \
   | jq

az group delete --name <your resource group name> --yes

# configure web app maven deployment
mvn com.microsoft.azure:azure-webapp-maven-plugin:1.12.0:config

# deploy with proxy (ref: https://github.com/microsoft/azure-maven-plugins/issues/520)
mvn package com.microsoft.azure:azure-webapp-maven-plugin:1.12.0:deploy -DproxySet=true -DproxyHost=sub.proxy.att.com -DproxyPort=8080
```

==

Deploy Spring Boot microservices to Azure Spring Cloud

```
az --version

az login # Sign into an azure account
az account show # See the currently signed-in account.

az account set --subscription <SUBSCRIPTION_ID>

# disable the certificate check across the CLI to work around "local issuer certificate" issue (ref to https://jiasli.github.io/azure-notes/cli/cli-proxy.html)
set AZURE_CLI_DISABLE_CONNECTION_VERIFICATION=1
#  install the spring-cloud extension for Azure CLI
az extension add -n spring-cloud -y    # need to run as administrator
```

```
# create a Azure Spring Cloud instance
az spring-cloud create \
    -g "$RESOURCE_GROUP_NAME" \
    -n "$SPRING_CLOUD_NAME" \
    --sku standard \
    --enable-java-agent

# set defaults
az configure --defaults group=${RESOURCE_GROUP_NAME}
az configure --defaults spring-cloud=${SPRING_CLOUD_NAME}

# create a service application in Azure Spring Cloud instance
az spring-cloud app create --name todo-service --resource-group "$RESOURCE_GROUP_NAME" --service "$SPRING_CLOUD_NAME"

# create MySQL
az mysql server create \
    --name ${SPRING_CLOUD_NAME}-mysql \
    --resource-group "$RESOURCE_GROUP_NAME" \
    --sku-name B_Gen5_1 \
    --storage-size 5120 \
    --admin-user "spring"

# create a todos database in MySQL server
az mysql db create \
    --name "todos" \
    --server-name ${SPRING_CLOUD_NAME}-mysql

# set up firewall rule
az mysql server firewall-rule create \
    --name ${SPRING_CLOUD_NAME}-mysql-allow-azure-ip \
    --resource-group "$RESOURCE_GROUP_NAME" \
    --server ${SPRING_CLOUD_NAME}-mysql \
    --start-ip-address "0.0.0.0" \
    --end-ip-address "0.0.0.0"

# deploy microservice
az spring-cloud app deploy --name todo-service --service "$SPRING_CLOUD_NAME" --resource-group "$RESOURCE_GROUP_NAME" --jar-path target/demo-0.0.1-SNAPSHOT.jar

# check the logs of the application (403?)
az spring-cloud app logs --name todo-service --service "$SPRING_CLOUD_NAME" --resource-group "$RESOURCE_GROUP_NAME" -f
```

==

Build a real-time event-driven Java solution

```
az group create \
    --name $RESOURCE_GROUP \
    --location $LOCATION

# create and configure an event hub
az eventhubs namespace create \
    --resource-group $RESOURCE_GROUP \
    --name $EVENT_HUB_NAMESPACE
az eventhubs eventhub create \
    --resource-group $RESOURCE_GROUP \
    --name $EVENT_HUB_NAME \
    --namespace-name $EVENT_HUB_NAMESPACE \
    --message-retention 1
az eventhubs eventhub authorization-rule create \
    --resource-group $RESOURCE_GROUP \
    --name $EVENT_HUB_AUTHORIZATION_RULE \
    --eventhub-name $EVENT_HUB_NAME \
    --namespace-name $EVENT_HUB_NAMESPACE \
    --rights Listen Send

# create Azure function
az storage account create \
    --resource-group $RESOURCE_GROUP \
    --name $STORAGE_ACCOUNT"p" \
    --sku Standard_LRS
az functionapp create \
    --resource-group $RESOURCE_GROUP \
    --name $FUNCTION_APP"-p"\
    --storage-account $STORAGE_ACCOUNT"p" \
    --consumption-plan-location $LOCATION \
    --runtime java \
    --functions-version 3
```

```
# retrieve connection strings for the storage account and the event hub
AZURE_WEB_JOBS_STORAGE=$( \
    az storage account show-connection-string \
        --resource-group $RESOURCE_GROUP \
        --name $STORAGE_ACCOUNT"p" \
        --query connectionString \
        --output tsv)
echo $AZURE_WEB_JOBS_STORAGE
EVENT_HUB_CONNECTION_STRING=$( \
    az eventhubs eventhub authorization-rule keys list \
        --resource-group $RESOURCE_GROUP \
        --name $EVENT_HUB_AUTHORIZATION_RULE \
        --eventhub-name $EVENT_HUB_NAME \
        --namespace-name $EVENT_HUB_NAMESPACE \
        --query primaryConnectionString \
        --output tsv)
echo $EVENT_HUB_CONNECTION_STRING

# store the connection strings in the application settings of Azure Function account
az functionapp config appsettings set \
    --resource-group $RESOURCE_GROUP \
    --name $FUNCTION_APP"-p" \
    --settings \
        AzureWebJobsStorage=$AZURE_WEB_JOBS_STORAGE \
        EventHubConnectionString=$EVENT_HUB_CONNECTION_STRING

# create local functions project with Maven
mvn archetype:generate --batch-mode \
    -DarchetypeGroupId=com.microsoft.azure \
    -DarchetypeArtifactId=azure-functions-archetype \
    -DappName=$FUNCTION_APP"-p" \
    -DresourceGroup=$RESOURCE_GROUP \
    -DappRegion=$LOCATION \
    -DappServicePlanName=$LOCATION"plan" \
    -DgroupId=com.learn \
    -DartifactId=telemetry-functions-producer

# retrieve local settings
func azure functionapp fetch-app-settings $FUNCTION_APP"-p"
```

==

*Add logging & monitoring

```
# get real-time log streaming from the app for basic troubleshooting
az webapp log tail -n <function app name> -g <resource group name>
func azure functionapp logstream <APP_NAME>
```

https://docs.microsoft.com/en-us/learn/modules/develop-azure-functions-app-with-maven-plugin/9-exercise-add-logging-to-azure-function

==

```
az account set --subscription {your subscription ID}

az configure --defaults group=[sandbox resource group name]

az group create --name <destination resource group name> --location <location name>

# retrieve the region where the resource group is located
az group show --name <group name> | jq -r '.location'

# get real-time log streaming from the app for basic troubleshooting
az webapp log tail -n <function app name> -g <resource group name>

yourResource=$(az resource show --resource-group <resource group name> --name <resource name> --resource-type <resource type> --query id --output tsv)

az resource move --destination-group <destination resource group name> --ids $yourResource

az resource list --resource-group <destination resource group name> --query [].type --output tsv | uniq

az acr build --registry <container_registry_name> --image webimage .

az acr task create --registry <container_registry_name> --name buildwebapp --image webimage --context https://github.com/MicrosoftDocs/mslearn-deploy-run-container-app-service.git --file Dockerfile --git-access-token <access_token>

export LOCATION=$(az group show --name $RESOURCEGROUP | jq -r '.location')

az sql server create \
    --name $SERVERNAME \
    --resource-group $RESOURCEGROUP \
    --location $LOCATION \
    --admin-user $ADMINLOGIN \
    --admin-password $PASSWORD

az sql db create --resource-group $RESOURCEGROUP \
    --server $SERVERNAME \
    --name marketplaceDb \
    --sample-name AdventureWorksLT \
    --service-objective Basic

az sql db show-connection-string --client sqlcmd --name marketplaceDb --server $SERVERNAME | jq -r

az vm create \
  --resource-group $RESOURCEGROUP \
  --name appServer \
  --image UbuntuLTS \
  --size Standard_DS2_v2 \
  --generate-ssh-keys
```