Maintenance Automation for Azure SQL databases

Author: Eric Rouach

Date: 2022-02-24

The following will guide you into setting up resources in Azure to execute any T-SQL command in all Azure SQL databases within a given Azure SQL server.

Steps:

1. Create an Azure Function App and as many functions as you need.
2. Create an Azure Data Factory and as many pipelines as functions.
3. Create a trigger to execute your Azure Data Factory Pipelines to automate your tasks.

Create an Azure Function App:

Open your web browser and go to the Azure Portal: <https://portal.azure.com/>

Make sure you use your customer’s directory.

Create a new Azure Function App:

Graphical user interface, text, application, Word

Description automatically generated

Create a new resource group (preferably including the name “DBSmart”) and choose a meaningful name for the App like “DBSmart-Function-App”.

Fill-in the Publish, Runtime stack, Version and Region fields as following:

Graphical user interface, text, application

Description automatically generated

In the Hosting tab, create a new storage account and select the relevant operating system:

Graphical user interface, text, application, email

Description automatically generated

Click Review + Create and wait for the resource deployment. Once the deployment has completed, clck “Go to resource”. This should take you the Function App’s Overview.

From the leftmost pane, select App files:

Graphical user interface, application

Description automatically generated

From the drop-down menu select the requirements file:

Graphical user interface, application

Description automatically generated

Add the following line to enable the use of the SQL Server module:

Graphical user interface, text, application, email

Description automatically generated

Save your changes.

Now, you need to create a function. Click on “Functions” from the leftmost pane:

Graphical user interface, text, application, email

Description automatically generated

Then click “Create”

Graphical user interface, application

Description automatically generated

Choose the following options:

Graphical user interface, application

Description automatically generated

Scroll Down and rename your function according to what it is doing:

Graphical user interface, text, application, email

Description automatically generated

Click “Create”.

Then, from the function page, select “Code + Test”:

Graphical user interface, text, application, email

Description automatically generated

Replace the appearing code with the following PowerShell code (don’t forget to insert your own T-SQL command at the designated place):

using namespace System.Net

# Input bindings are passed in via param block.

param($Request, $TriggerMetadata)

# Write to the Azure Functions log stream.

Write-Host "PowerShell HTTP trigger function processed a request."

#Set Variables:

$UserName = $Request.Body.UserName # 🡸 Will be defined in the ADF settings / body

$Password = $Request.Body.Password # 🡸 Will be defined in the ADF settings / body

$ServerInstance = 'adf-demo-eric.database.windows.net' # <=== Insert your Server Name

$Database = 'master'

$Query = 'SELECT name FROM sys.databases WHERE database\_id > 1'

$TargetTenants = Invoke-Sqlcmd -ServerInstance $ServerInstance -Database $Database -UserName $UserName -Password $Password -Query $Query

$Query = 'create table dbo.Test555 (Id int)' # 🡸 Insert your command here

#Set a foreach loop for each of the Target Tenants:

$TargetTenants | ForEach-Object{

$TargetTenant = $\_.name

Invoke-Sqlcmd -ServerInstance $ServerInstance -Database $TargetTenant -UserName $UserName -Password $Password -Query $Query -Verbose

Write-Host "$Query is being executed in $TargetTenant database"

}

Click “Save”.

The next step is creating an Azure Data Factory. It will be executing a pipeline for every function your wish to run. Each pipeline will be execution a single “Azure Function” activity.

Back to Azure Portal’s home page, create a new Azure Data Factory:

Graphical user interface, application, Word

Description automatically generated

Fill-in the fields as following and go to the Git configuration tab where you will check the “Configure Git later” checkbox:

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Move on to the “Review + Create” tab and make sure the validation passes successfully. Then, click “Create”:

Graphical user interface, text, application, email

Description automatically generated

The deployment should take a few moments; once completed, go to the resource.

From your newly created ADF, open the ADF Studio:

Graphical user interface, application

Description automatically generated

From ADF Studio’s home page, choose to create a new pipeline:

Graphical user interface, text, application

Description automatically generated

In the properties pane, rename your pipeline according to what the Azure Function is doing:

Graphical user interface, text, application, email

Description automatically generated

Close the Properties pane by clicking the properties button:

Graphical user interface, text, application

Description automatically generated

From the Activities pane, select and drag an “Azure Function” activity and rename it accordingly:

Graphical user interface, text, application

Description automatically generated

From the settings tab, create anew Azure Function Linked Service:

Graphical user interface, text, application, email

Description automatically generated

Fill-in the fields as following:

Graphical user interface, text, application

Description automatically generated

You may find the Function Key from the function’s page:

Graphical user interface, text, application

Description automatically generated

Still in ADF’s settings tab, copy and paste the function name and choose the POST method.

In the Body field, enter your Azure SQL Server’s UserName and Password as following:

{"UserName":"yourusername","Password":"YourP@ssrord!"}

Make sure to spell the “UserName” and “Password” properties so they match the variables names in the function’s PowerShell code!!!

Your settings tab should look like this:

Graphical user interface

Description automatically generated

Validate your pipeline to make sure no errors are found in the ADF pipeline configuration:

Graphical user interface, application

Description automatically generated

Then click “Debug” to try your pipeline.

Graphical user interface, application

Description automatically generated

Make sure the pipeline runs successfully and executes what the function is supposed to do and publish (save) its changes:

Graphical user interface, application, Word

Description automatically generated

The final (and easiest) step is the creation of a trigger so that your pipeline can run automatically.

Add a new Trigger:

Graphical user interface, text, application

Description automatically generated

Fill-in the fields as requested and click “Ok”:

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

This is it!