Use GitHub Actions to connect to Azure SQL Database

Article • 02/16/2023 • 5 minutes to read

Get started with GitHub Actions by using a workflow to deploy database updates to Azure SQL Database.

Prerequisites

You need:

- An Azure account with an active subscription. Create an account for free .
- A GitHub repository with a dacpac package (`Database.dacpac`). If you don't have a GitHub account, sign up for free .
- An Azure SQL Database.
  - Quickstart: Create an Azure SQL Database single database
  - How to create a dacpac package from your existing SQL Server Database

Workflow file overview

A GitHub Actions workflow is defined by a YAML (.yml) file in the `/.github/workflows/` path in your repository. This definition contains the various steps and parameters that make up the workflow.

The file has two sections:

Section	Tasks
Authentication	1.1. Generate deployment credentials.
Deploy	1. Deploy the database.

Generate deployment credentials

Service principal

Create a service principal with the az ad sp create-for-rbac command in the Azure CLI. Run this command with Azure Cloud Shell in the Azure portal or by selecting the **Try it** button.

Azure CLI

```
az ad sp create-for-rbac --name "myML" --role contributor \
                            --scopes /subscriptions/<subscription-
```

```
id>/resourceGroups/<group-name> \
                            --sdk-auth
```

In the example above, replace the placeholders with your subscription ID, resource group name, and app name. The output is a JSON object with the role assignment credentials that provide access to your App Service app similar to below. Copy this JSON object for later.

Output

```
  {
    "clientId": "<GUID>",
    "clientSecret": "<GUID>",
    "subscriptionId": "<GUID>",
    "tenantId": "<GUID>",
    (...)
  }
```

Copy the SQL connection string

In the Azure portal, go to your Azure SQL Database and open **Settings** > **Connection strings**. Copy the **ADO.NET** connection string. Replace the placeholder values for `your_database` and `your_password`. The connection string looks similar to this output.

Output

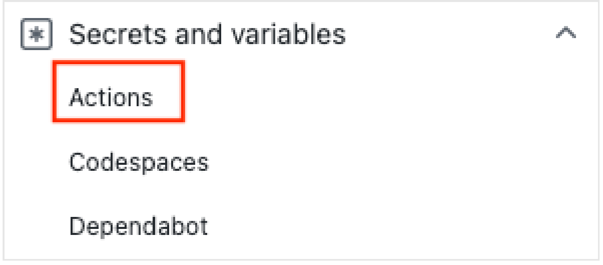
```
    Server=tcp:my-sql-server.database.windows.net,1433;Initial Catalog={your-
database};Persist Security Info=False;User ID={admin-name};Password={your-
password};MultipleActiveResultSets=False;Encrypt=True;TrustServerCertificate=False;Co
nnection Timeout=30;
```

Use the connection string as a GitHub secret.

Configure the GitHub secrets

Service principal

1. In GitHub, go to your repository.
2. Select **Security** > **Secrets and variables** > **Actions**.



3. Select **New repository secret**.

4. Paste the entire JSON output from the Azure CLI command into the secret's value field. Give the secret the name `AZURE_CREDENTIALS`.

5. Select **Add secret**.

Add your workflow

1. Go to **Actions** for your GitHub repository.

2. Select **Set up your workflow yourself**.

3. Delete everything after the `on:` section of your workflow file. For example, your remaining workflow may look like this.

YAML

```yaml
name: CI

on:
push:
    branches: [ main ]
pull_request:
    branches: [ main ]
```

4. Rename your workflow `SQL for GitHub Actions` and add the checkout and login actions. These actions check out your site code and authenticate with Azure using the `AZURE_CREDENTIALS` GitHub secret you created earlier.

Service principal

YAML

```yaml
name: SQL for GitHub Actions

on:
push:
    branches: [ main ]
pull_request:
    branches: [ main ]
```

```yaml
jobs:
build:
    runs-on: windows-latest
    steps:
    - uses: actions/checkout@v1
    - uses: azure/login@v1
    with:
        creds: ${{ secrets.AZURE_CREDENTIALS }}
```

1. Use the Azure SQL Deploy action to connect to your SQL instance. You should have a dacpac package (`Database.dacpac`) at the root level of your repository.

YAML

```yaml
- uses: azure/sql-action@v2
  with:
    connection-string: ${{ secrets.AZURE_SQL_CONNECTION_STRING }}
    path: './Database.dacpac'
    action: 'Publish'
```

2. Complete your workflow by adding an action to logout of Azure. Here's the completed workflow. The file appears in the `.github/workflows` folder of your repository.

Service principal

YAML

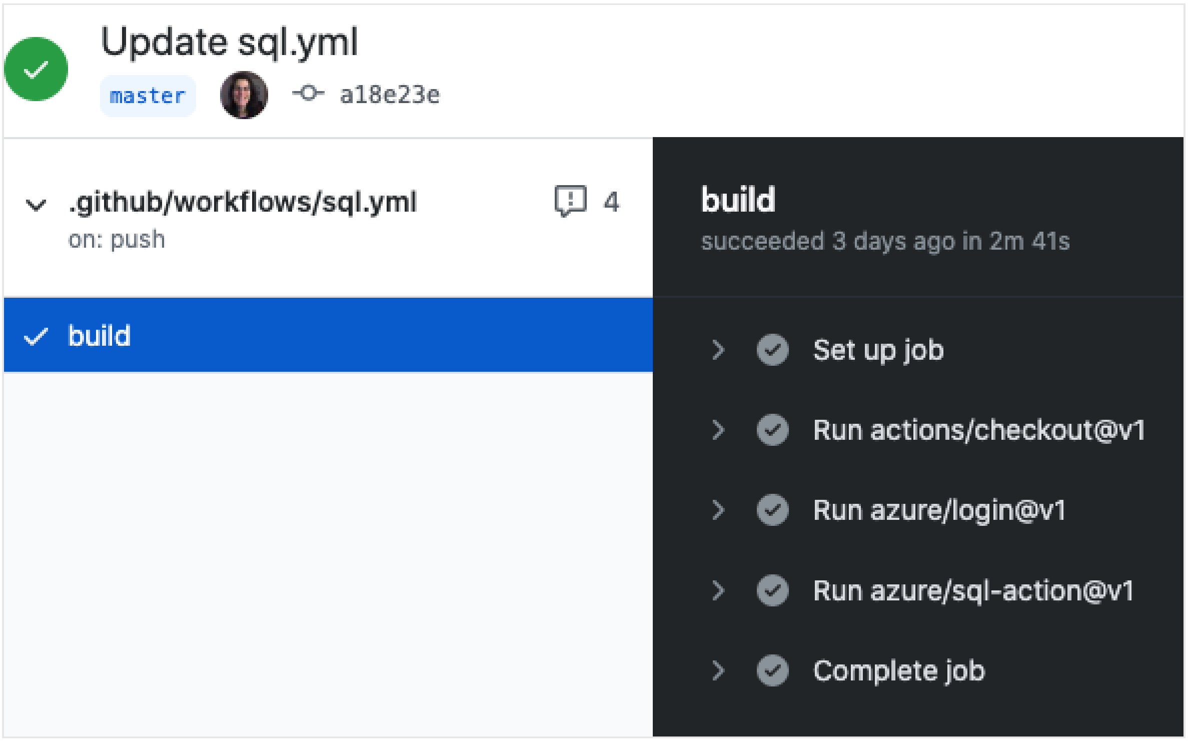
```yaml
name: SQL for GitHub Actions

on:
push:
    branches: [ main ]
pull_request:
    branches: [ main ]


jobs:
build:
    runs-on: windows-latest
    steps:
    - uses: actions/checkout@v1
    - uses: azure/login@v1
    with:
        creds: ${{ secrets.AZURE_CREDENTIALS }}

- uses: azure/sql-action@v2
  with:
    connection-string: ${{ secrets.AZURE_SQL_CONNECTION_STRING }}
    path: './Database.dacpac'
    action: 'Publish'
```

```
      # Azure logout
  - name: logout
    run: |
       az logout
```

Review your deployment

1. Go to **Actions** for your GitHub repository.

2. Open the first result to see detailed logs of your workflow's run.



Update sql.yml
master a18e23e
.github/workflows/sql.yml 4
on: push
build
build
succeeded 3 days ago in 2m 41s
Set up job
Run actions/checkout@v1
Run azure/login@v1
Run azure/sql-action@v1
Complete job

Clean up resources

When your Azure SQL database and repository are no longer needed, clean up the resources you deployed by deleting the resource group and your GitHub repository.

Next steps

Learn about Azure and GitHub integration