Automating Deployments with Azure DevOps

Erin Dempster
5 Mar 2024

Erin Dempster

She/Her

Microsoft® Most Valuable Professional

Data Operations Team Lead

Trean Corporation

- SQL Server DBA
- Azure Administrator
- Azure DevOps Administrator

https://www.erindempster.com

@em_dempster

https://www.linkedin.com/in/erindempster

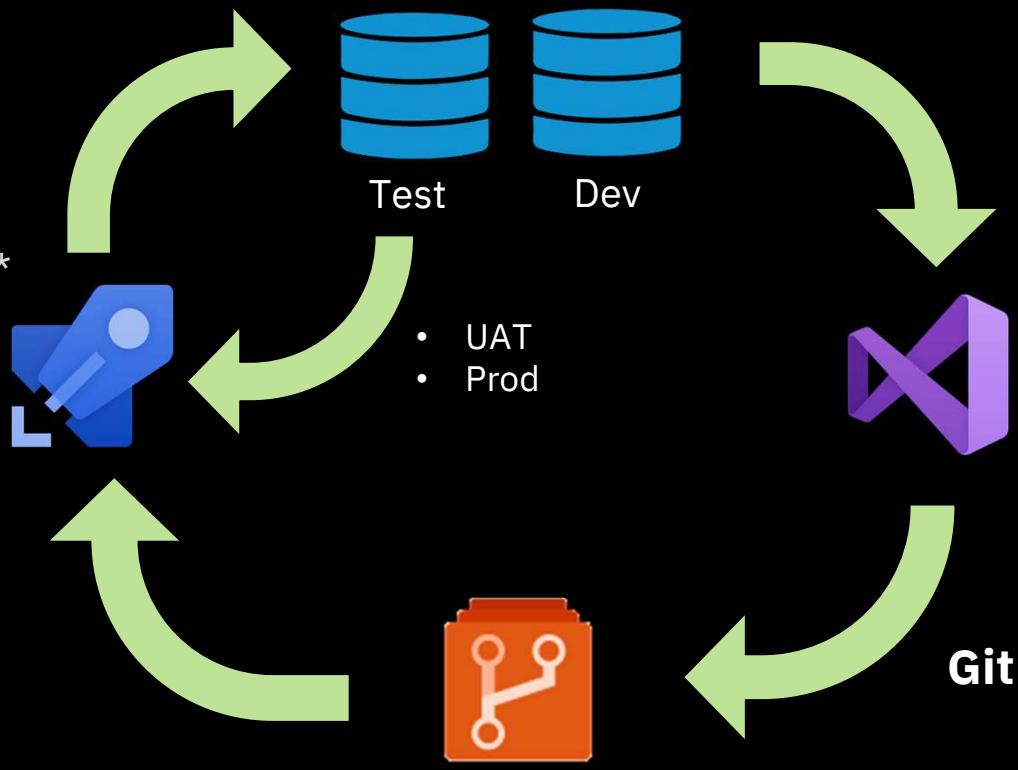
Speaker – PASS Summit, SQL Saturdays +

Author – SQLServerCentral.com

Agenda

Azure Pipelines

1. Build
2. Automated Tests*
3. Approve
4. Deploy
5. Repeat 3&4



Visual Studio

- Reverse Engineer
- Develop
- Unit Test
- Commit Changes

Git Client

Azure Repos

Git Repositories

Sample Code

Coming Soon

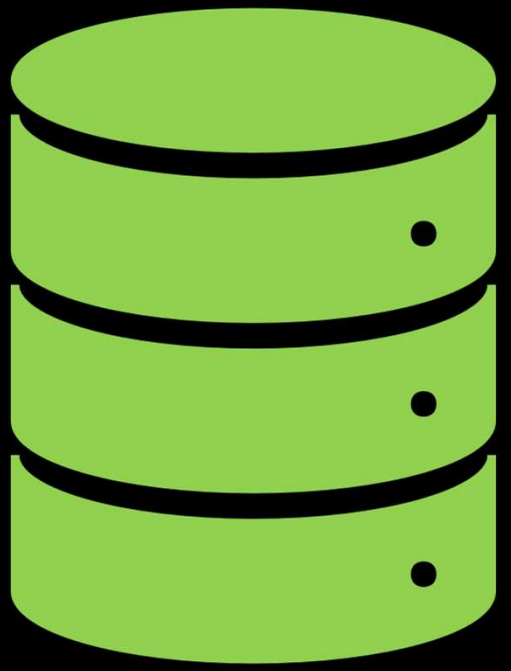
https://github.com/TrueNorthMN/DevOps

Introduction to Visual Studio

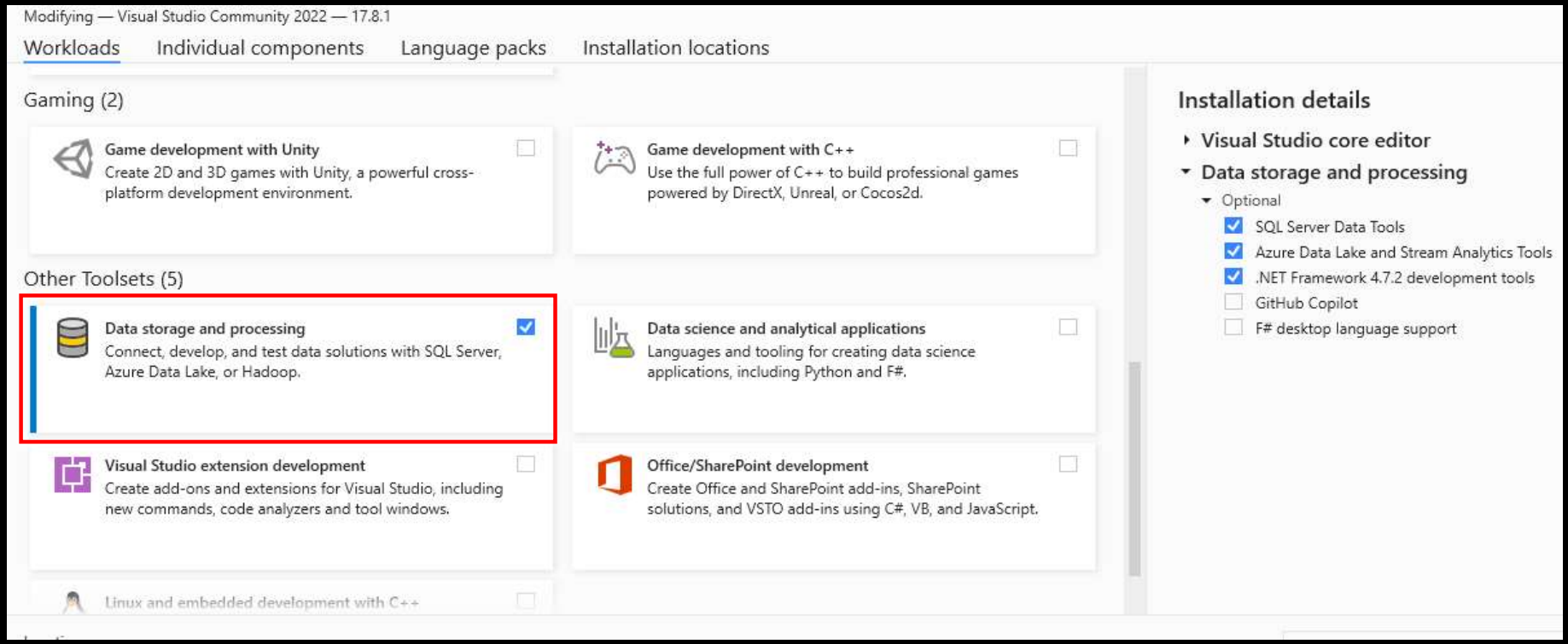
SQL Database Project

Visual Studio – SQL Database Project

- Long-Time Support for SQL Databases
- Build Projects with msbuild.exe
- Creates .DACPAC for Deployment
- Define Deployment Settings

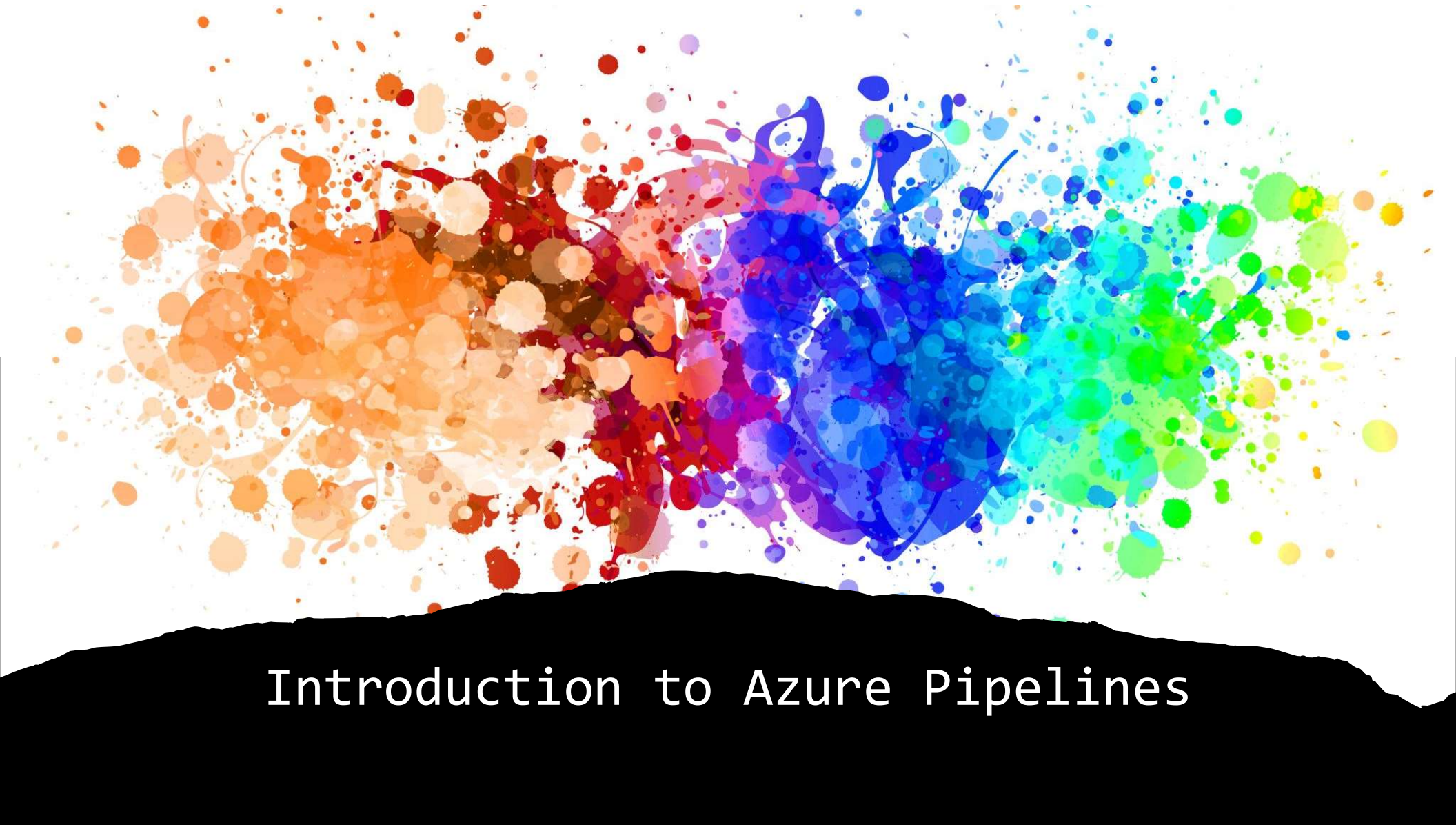


Visual Studio Installer



Demo

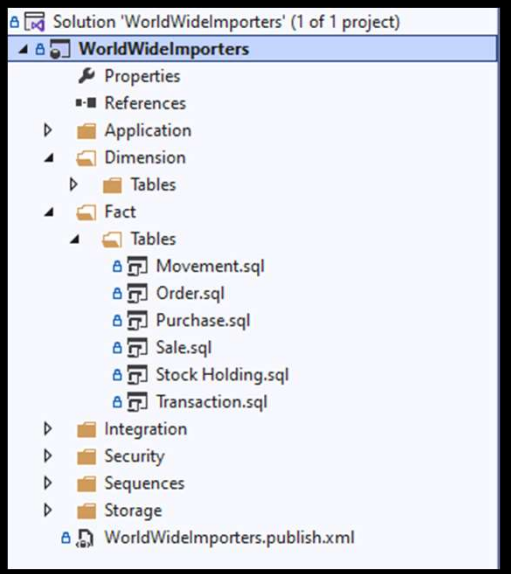
Visual Studio and Azure DevOps Overview


Introduction to Azure Pipelines

Azure Pipelines

- Manages code builds and deployments

- Triggered by source code check-ins

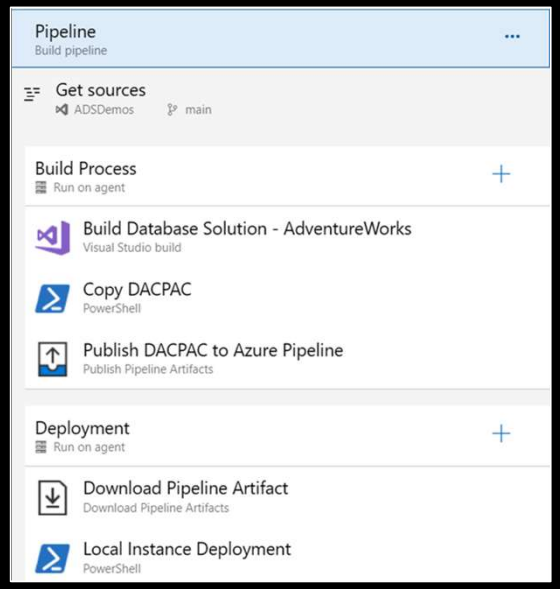


Azure Pipelines

- Many pre-defined tasks
  - Visual Studio Build
  - Publish & Download Artifacts
- Scripting tasks
  - PowerShell
  - BASH

Define **YOUR** workflow in YAML

```yaml
trigger:
- main

pool:
  vmImage: windows-latest

stages:
- stage: buildProcess
  jobs:
  - job: buildDACPAC
    displayName: Build Database DACPAC
    steps:
    - task: VSBuild@1
      inputs:
        solution: '**\AdventureWorks.sln'
    - task: PublishPipelineArtifact@1
      inputs:
        targetPath: '$(Build.SourcesDirectory)\AdventureWorks
        artifact: 'Databases'
        publishLocation: 'pipeline'
```



Ways to Run Pipelines

- Azure-hosted Pipeline Agents
  - Great for building code/build validation
- Self-hosted Pipeline Agents
  - Preferred for deployments

Pipeline Options

	Build Pipeline	Release Pipeline
Classic	No*	Yes
YAML	Yes	Yes

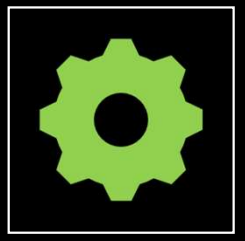
Optimal Design - Multi-stage YAML Build and Release Pipeline

Azure Pipelines

- YAML Schema - https://tinyurl.com/4z7x3mkw
- Task Index - https://tinyurl.com/56vf436z
- Pipeline Variables - https://tinyurl.com/ye2ypvum

What is YAML?

"**YAML Ain't M**arkup **L**anguage"



Configuration Layout

Used by both Azure DevOps and GitHub Actions

YAML - Azure Pipelines vs GitHub Actions

```yaml
trigger:
- main

pool:
  vmImage: windows-latest

jobs:
- job: buildDACPAC
  displayName: Build Azure Data Studio DACPAC
  steps:
  Settings
  - task: DotNetCoreCLI@2
    inputs:
      command: 'build'
      projects: '**/Adventureworks.sqlproj'
```

```yaml
name: Build AdventureWorks

on:
  push:
    branches: [main]
  workflow_dispatch:

jobs:
  build:
    runs-on: windows-latest
    steps:
    - uses: actions/checkout@v3

    - name: Add MSBuild to PATH
      uses: microsoft/setup-msbuild@v1.0.2

    - name: Build
      working-directory: ${{env.GITHUB_WORKSPACE}}
      run: msbuild /m /p:Configuration=Release Adventureworks/Adventureworks.sqlproj
```

Configuration Layout

Hierarchical

<Pipeline>

- Stages
  - Jobs
    - Steps
      - Task

Basic Pipeline

```yaml
trigger:
- main

pool:
  vmImage: windows-latest

steps:
Settings
- task: DotNetCoreCLI@2
  inputs:
    command: 'build'
    projects: '**/Adventureworks.sqlproj'
Settings
- task: SqlAzureDacpacDeployment@1
  inputs:
    azureSubscription: 'MVP Community'
    AuthenticationType: 'servicePrincipal'
    ServerName: '
    DatabaseName: 'AdventureWorks'
    deployType: 'DacpacTask'
    DeploymentAction: 'Publish'
    DacpacFile: 'Adventureworks.dacpac'
    IpDetectionMethod: 'AutoDetect'
```

Implied Single Stage with one job

1st task – builds the project

2nd task – deploys the DACPAC using SQLPackage.exe

Both tasks run in the same agent instance

When to use this scenario?

- Proof of concept
- Testing connections
- Build Validation/Automated Testing

Explicit Job Definitions

```yaml
trigger:
- main

pool:
  vmImage: windows-latest

jobs:
- job: buildDACPAC
  displayName: Build Azure Data Studio DACPAC
  steps:
  Settings
  - task: DotNetCoreCLI@2
    inputs:
      command: 'build'
      projects: '**/Adventureworks.sqlproj'
- job: deployDACPAC
  displayName: Deploy DACPAC to Azure SQL DB
  steps:
  Settings
  - task: SqlAzureDacpacDeployment@1
    inputs:
      azureSubscription: 'MVP Community'
      AuthenticationType: 'servicePrincipal'
      ServerName: 'endlessreporting.database.windows.net'
      DatabaseName: 'AdventureWorks'
      deployType: 'DacpacTask'
      DeploymentAction: 'Publish'
      DacpacFile: 'Adventureworks.dacpac'
      IpDetectionMethod: 'AutoDetect'
```

Two Jobs each with one task

1st job – builds the project

2nd job – deploys the DACPAC using SQLPackage.exe

Each job runs in a separate agent instance

Explicit Job Definitions

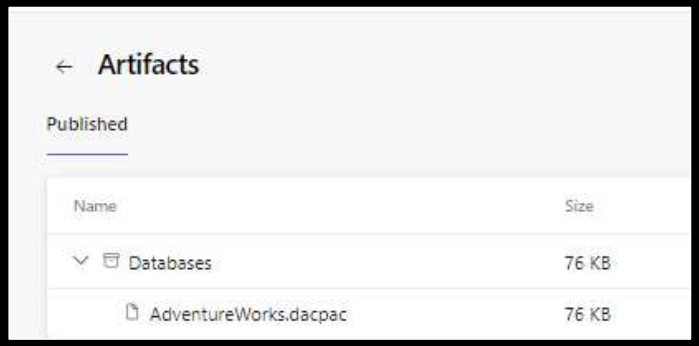
```yaml
trigger:
- main

pool:
  vmImage: windows-latest

jobs:
- job: buildDACPAC
  displayName: Build Azure Data Studio DACPAC
  steps:
  Settings
  - task: DotNetCoreCLI@2
    inputs:
      command: 'build'
      projects: '**/Adventureworks.sqlproj'
- job: deployDACPAC
  displayName: Deploy DACPAC to Azure SQL DB
  steps:
  Settings
  - task: SqlAzureDacpacDeployment@1
    inputs:
      azureSubscription: 'MVP Community'
      AuthenticationType: 'servicePrincipal'
      ServerName: 'endlessreporting.database.windows.net'
      DatabaseName: 'AdventureWorks'
      deployType: 'DacpacTask'
      DeploymentAction: 'Publish'
      DacpacFile: 'Adventureworks.dacpac'
      IpDetectionMethod: 'AutoDetect'
```

There is a problem with this pipeline!

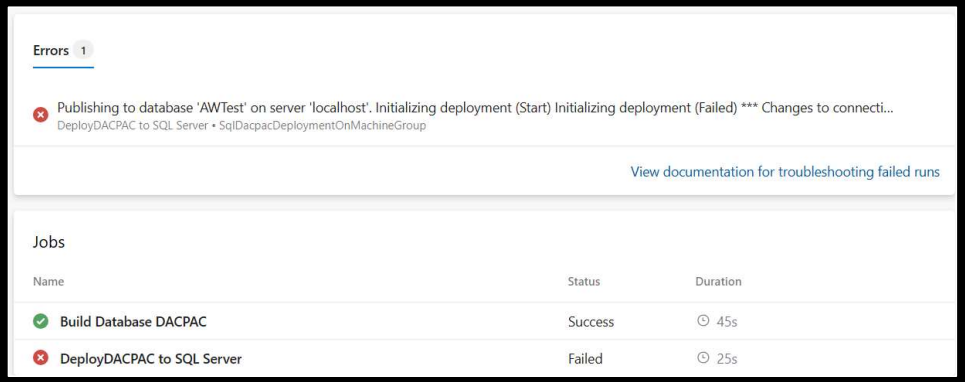
DACPAC is not shared between jobs

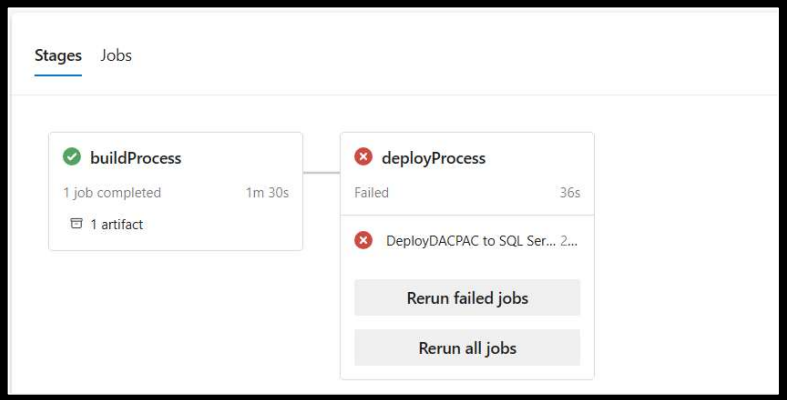
Publish DACPAC as a Pipeline Artifact

```yaml
- job: buildDACPAC
  displayName: Build Azure Data Studio DACPAC
  steps:
  Settings
  - task: DotNetCoreCLI@2
    inputs:
      command: 'build'
      projects: '**/Adventureworks.sqlproj'
  Settings
  - task: PublishPipelineArtifact@1
    inputs:
      targetPath: '$(Pipeline.Workspace)\AdventureWorks\bin\debug\Adventureworks.dacpac'
      artifact: 'Databases'
      publishLocation: 'pipeline'
```



Explicit Stage Declaration



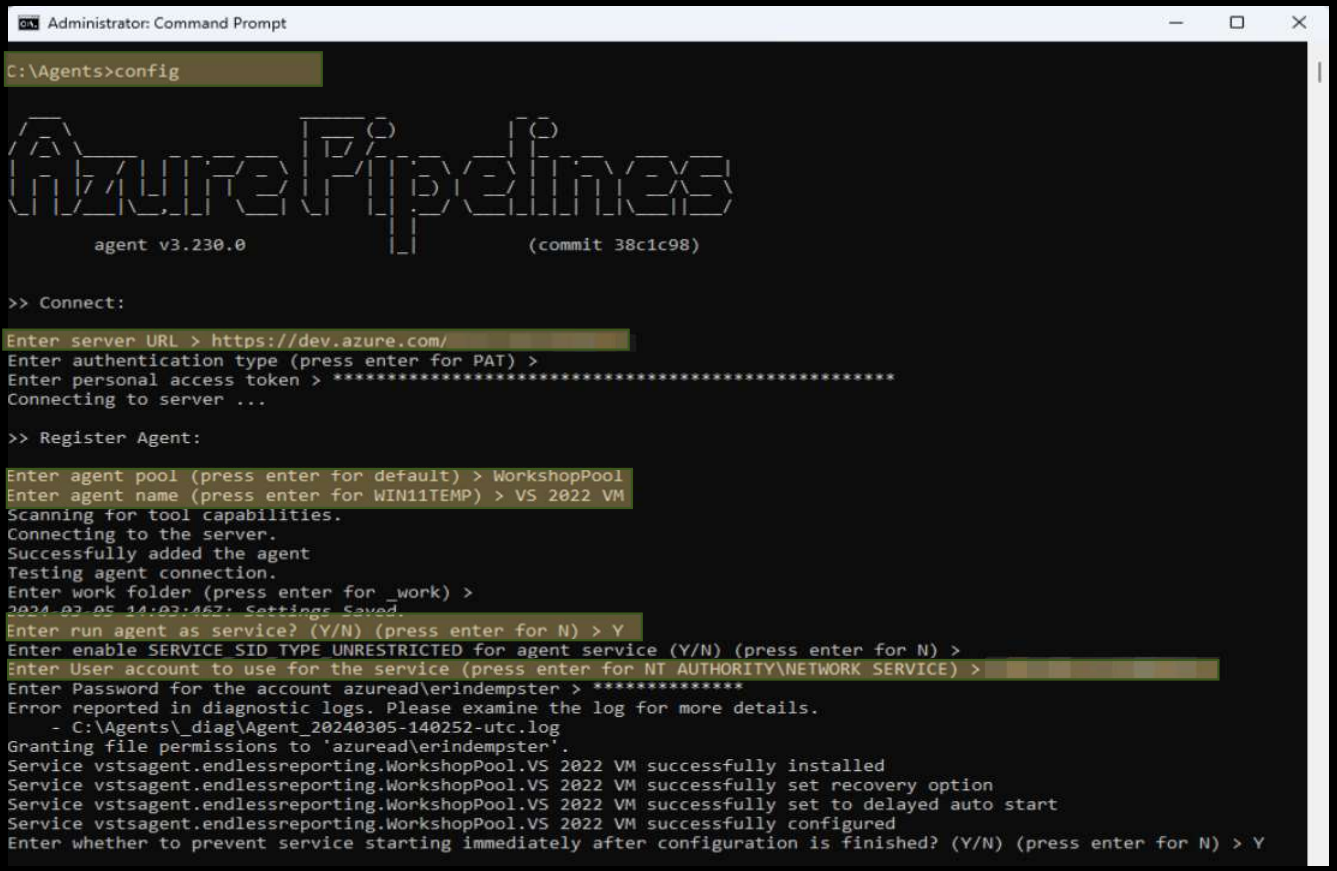



```yaml
trigger:
- main

pool:
  vmImage: windows-latest

stages:
- stage: buildProcess
  jobs:
  - job: buildDACPAC
    displayName: Build Database DACPAC
    steps:

- stage: deployProcess
  jobs:
  - job: deployDACPACToTest
    displayName: DeployDACPAC to SQL Server
    pool: WorkshopPool
    steps:
```

Agent Setup

- Define a Pipeline Agent Pool
- Generate a Personal Access Token (PAT)
- Install Agent
- Configure Agent

Self-Hosted Pipeline Agent




Common Database Tasks

Common Tasks for Databases

Building DACPACs

- VSBuild@1 – Build Visual Studio project with MSBuild
- DotNetCoreCLI@2 – Build Azure Data Studio project

Common Tasks for Databases

Artifact Management

- PublishBuildArtifacts@1 – Publish Files to Pipeline
- DownloadPipelineArtifact@2 – Download Files from Pipeline

Common Tasks for Databases

Deployment Tasks

- SqlAzureDacpacDeployment@1 – Deploy to Azure SQL DB
- AzureCLI@2 – Execute PowerShell scripts in Azure context
- PowerShell@2 – Execute PowerShell scripts

Deploy DACPAC with PowerShell

- Requires
  - SQLPackage.exe

```
$connectionString = "server=myserver;database=mydatabase;authentication=Active Directory MSI;
```

```
. 'C:\Program Files\Microsoft SQL Server\160\DAC\bin\sqlpackage.exe'
      /action:Publish
      /sourceFile:$($filePath)\$databaseName.dacpac
      /profile:$($filePath)\$databaseName.publish.xml
      /targetConnectionString:"$connectionString"
```

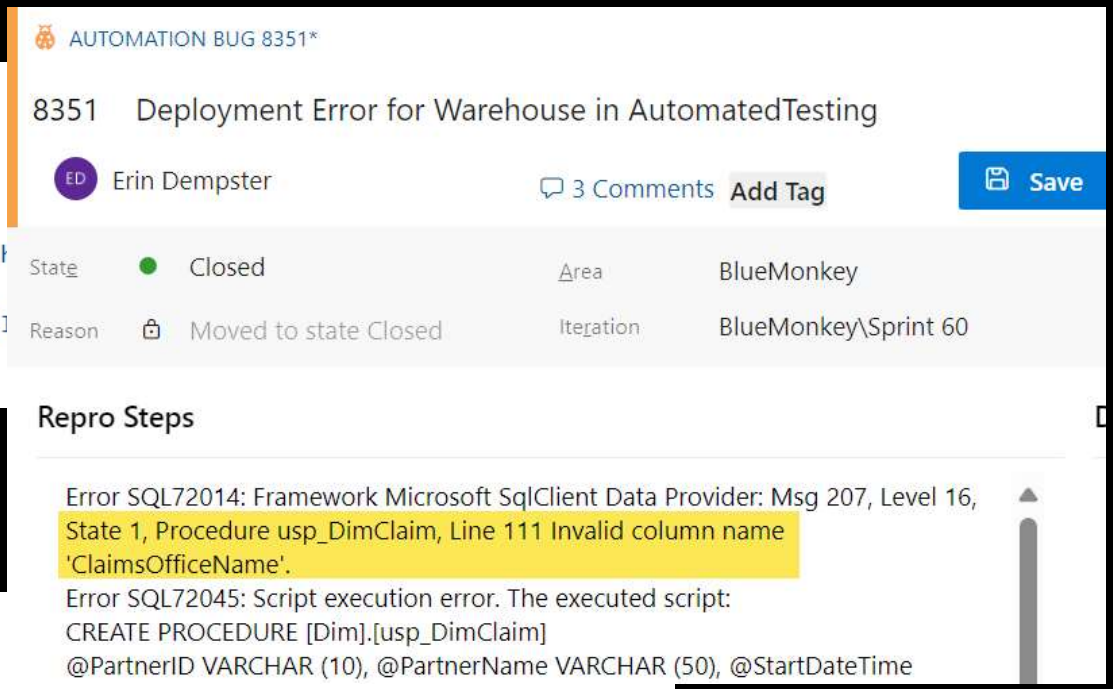
Deploy DACPAC with PowerShell

- Requires
  - SQLPackage.exe

```
$connectionString = "server=myserver;database=mydatabase;authentication=Active Directory MSI; encrypt=yes;TrustServerCertificate=no"

$dacOutput = . 'C:\Program Files\Microsoft SQL Server\160\DAC\bin\sqlpackage.exe'
                    /action:Publish
                    /sourceFile:$($filePath)\$databaseName.dacpac
                    /profile:$($filePath)\$databaseName.publish.xml
                    /targetConnectionString:"$connectionString"
     2>&1

$errors = @()
$logRemaining = $false

for ($i = 0; $i -lt $dacOutput.Count; $i++)
{
    #
    # SQLPackage.exe will fail on errors, so errors are towards the end of the output. Once errors start to appear,
    # grab the rest of the output lines to include additional information.
    #
    if ($dacOutput[$i].ToString().StartsWith("Error") -or $logRemaining -eq $true)
    {
        $errors += $dacOutput[$i]
        $logRemaining = $true
    }
}
```

Logging Bugs with PowerShell

```yaml
- task: AzureCLI@2
  displayName: Deploy Warehouse Database Changes
  inputs:
    scriptType: ps
    azureSubscription: $(azureSubscription)
    scriptPath: $(Build.SourcesDirectory)/PowerShell/DBDeployWith
    powerShellErrorActionPreference: continue
    arguments: -databaseName "Warehouse" -fullInstanceName $(SQL
  env:
    AZURE_DEVOPS_EXT_PAT: $(System.AccessToken)
```




```powershell
az extension add --name azure-devops

$assignBugTo = $env:Build_RequestedFor

az boards work-item create --title "Deployment Error for $databaseName in $environment"
                           --type "Automation Bug"
                           --project "$projectName"
                           --assigned-to $assignBugTo
                           --fields "Microsoft.VSTS.TCM.ReproSteps=$errorString"
                           --organization "https://dev.azure.com/<myOrg>/"
```

Connection Strings

Entra ID (Azure AD) Managed Identity

```
$connectionString = "server=myserver;database=mydatabase;authentication=Active Directory MSI;
```

Windows Authentication

```
$connectionstring = "server=myserver; database=mydatabase; Integrated Security=SSPI;"
```

Upcoming Sessions in Southeast US

- SQL Saturday Atlanta – April 19th & 20th
  - ½ Day Pre-Con – Automating Deployments with Azure DevOps
    - Friday 8 – noon
  - General Session – Intro to Automated Deployments with GitHub
- SQL Saturday Jacksonville – May 4th
  - Intro to Automated Deployments with Azure DevOps
  - Intro to Automated Deployments with GitHub