Kamil Nowiński

Microsoft Data Platform **MVP**

Speaker, blogger, data enthusiast

Group Manager at Avanade UK&I (www.avanade.com)

>20 yrs experience as DEV/BI/(DBA)

Member of the Data Community PL

Founder of blog SQLPlayer (www.SQLplayer.net)

GitHub: #adftools, SCD Merge Wizard and more...

SQL Server Certificates:

MCITP, MCP, MCTS, MCSA, MCSE Data Platform,

MCSE Data Management & Analytics, DevOps Expert

Moreover: Bicycle, Running, Digital photography

@NowinskiK, @SQLPlayer

- Technical posts
- Various skill level
- Cheet sheets
- Recommended books
- Many useful other links
- Interviews (Podcast)
- YouTube Channel: www.SQLPlayer.net/YouTube

SQL Player
Play with data & have fun!
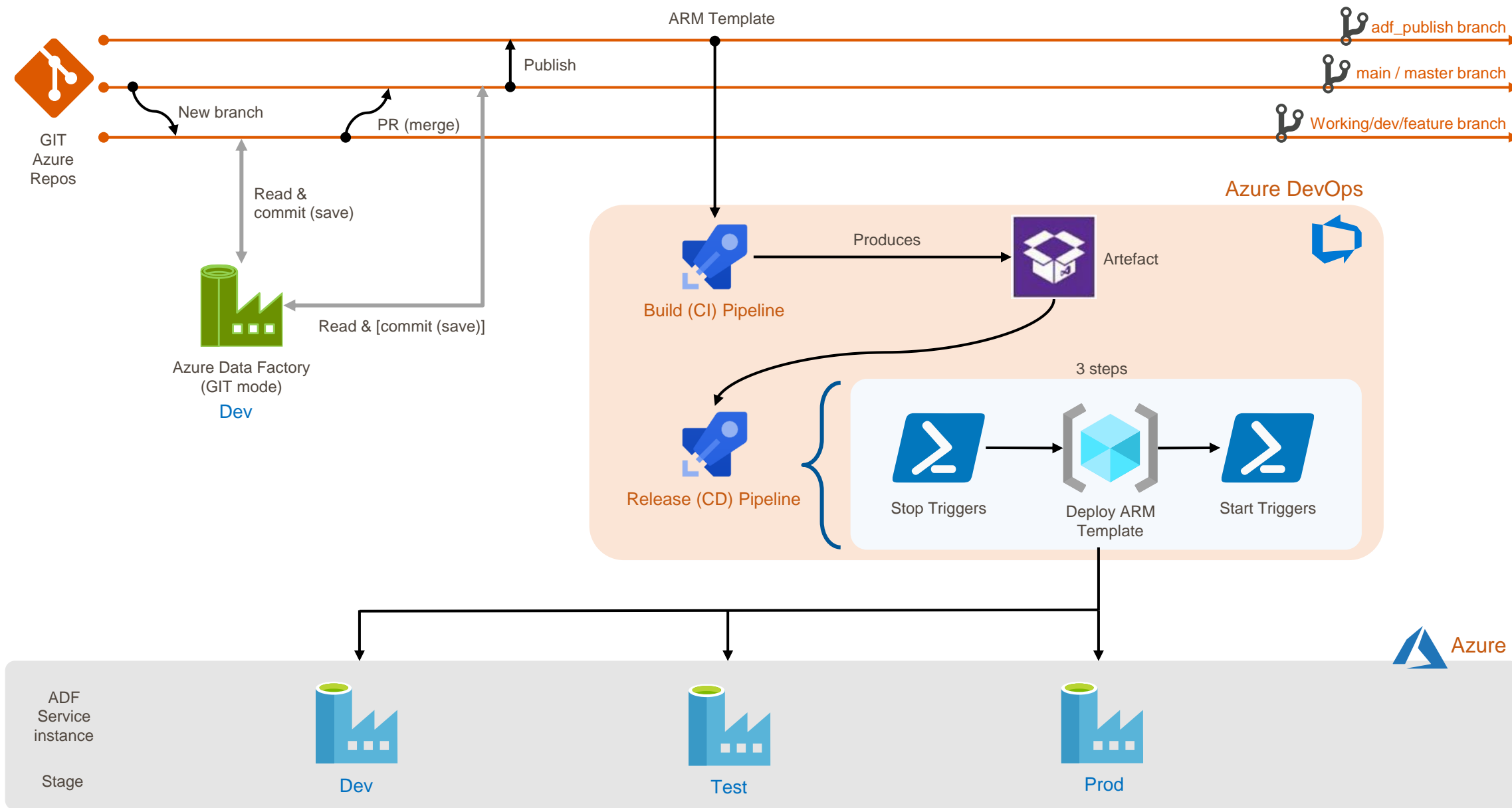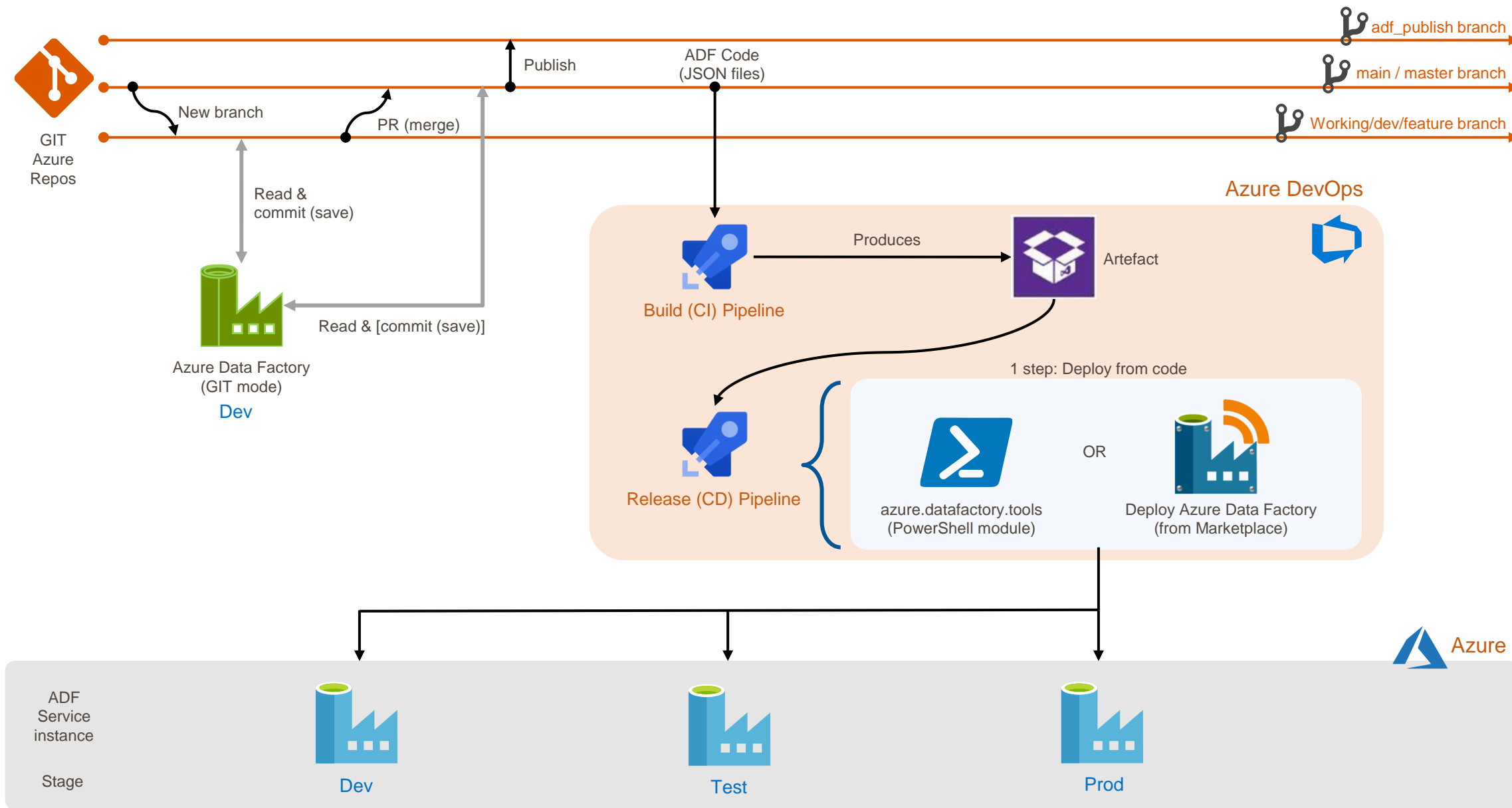
www.SQLPlayer.net

Scan me

github.com
/NowinskiK/CommunityEvents

- Azure Data Factory – DEPLOYMENT only

- Two (three?) methods of ADF deployment

- How these methods work

- Differences

- npm module from Microsoft – now you can fully automate CI (build)

- #adftools – make your life easier!

Deployment (1): Microsoft's method (ARM Template)

SQL Player
Play with data & have fun!

GIT
Azure
Repos

New branch

PR (merge)

ARM Template

Publish

adf_publish branch

main / master branch

Working/dev/feature branch

Read &
commit (save)

Read & [commit (save)]

Azure Data Factory
(GIT mode)

Dev

Azure DevOps

Build (CI) Pipeline

Produces

Artefact

3 steps

Release (CD) Pipeline

Stop Triggers

Deploy ARM
Template

Start Triggers

Azure

ADF
Service
instance

Stage

Dev

Test

Prod

Deployment (2): Directly from code method

SQL Player
Play with data & have fun!

adf_publish branch

main / master branch

Publish

ADF Code
(JSON files)

Working/dev/feature branch

GIT
Azure
Repos

New branch

PR (merge)

Read &
commit (save)

Azure DevOps

Produces

Artefact

Build (CI) Pipeline

Read & [commit (save)]

Azure Data Factory
(GIT mode)

Dev

1 step: Deploy from code

OR

Release (CD) Pipeline

azure.datafactory.tools
(PowerShell module)

Deploy Azure Data Factory
(from Marketplace)

Azure

ADF
Service
instance

Stage

Dev

Test

Prod

ADF – Currently available methods of deployment – pros & cons

ARM Template from "adf_publish" branch

- Faster, "incremental"

- Appears in „Deployments" for Resource Group

- Parametrize elements exposed within the ARM Template Parameter *

- Full ADF (all artefacts) can be deploy only

- Restriction of 256 parameters

- Limitation to one publish branch only (adf_publish)

- Manual "Publish" step

Before 2021

Rest-Api/PowerShell script from code (JSON objects)

- Slower

- Doesn't appear in „Deployments" for Resource Group

- Parameterize any artefact of the Data Factory

- Selectively deploy a subset of artefacts

- Eliminates an enforcement to use only one (adf_publish) branch if company's branches policy is much complex

🌐 Post: Two methods of deployment Azure Data Factory

ADF – ARM Template enhanced deployment method

ARM Template from "adf_publish" branch

- Faster, "incremental"

- Appears in „Deployments" for Resource Group

- Parametrize elements exposed within the ARM Template Parameter *

- Full ADF (all artefacts) can be deploy only

- Restriction of 256 parameters

- ~~Limitation to one publish branch only (adf_publish)~~

- ~~Manual "Publish" step~~

- Not user-friendly npm library

- Requires adding "package" file to a repo

Rest-Api/PowerShell script from code (JSON objects)

- Slower

- Doesn't appear in „Deployments" for Resource Group

- Parameterize any artefact of the Data Factory

- Selectively deploy a subset of artefacts

- Eliminates an enforcement to use only one (adf_publish) branch if company's branches policy is much complex

Since 2021

Just a few small things...

- **[Continuous integration and delivery](#)** in **Azure Data Factory**
- **[Automated publishing for continuous integration and delivery](#)**
- **[Use custom parameters with the Resource Manager template](#)**
- **[Sample pre- and post-deployment script](#)**
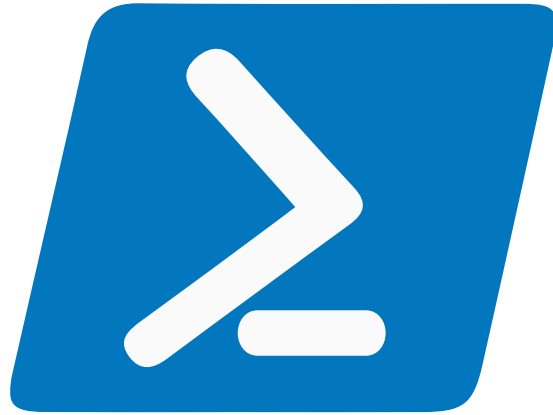
Automated publishing via CI/CD within npm & ARM Template
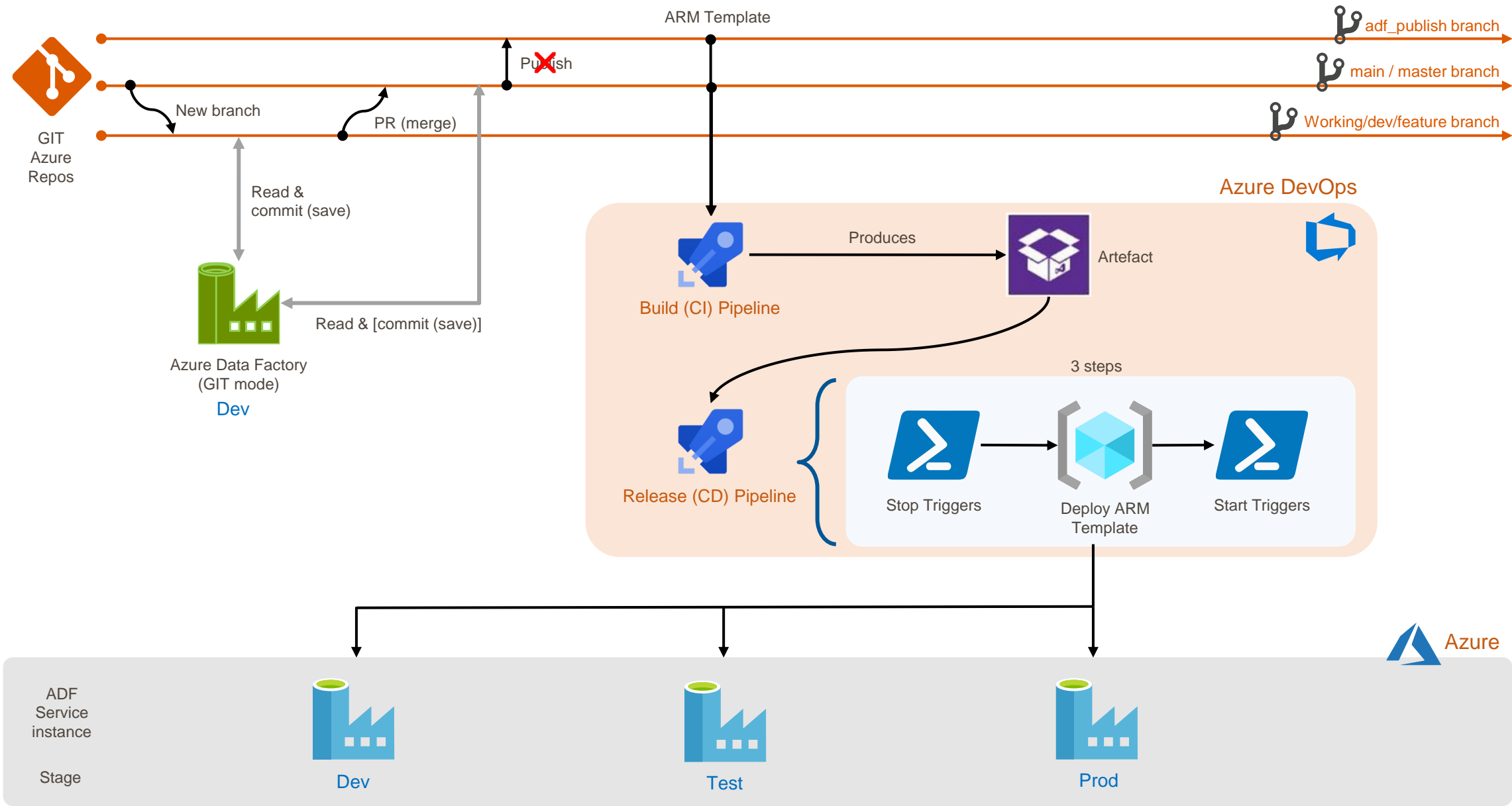
DEMO
... in a minute;
a question first...

SQL Player

DEMO #1
ARM Template
deployment

SQL Player
Play with data & have fun!

#adftools

#adftools

Two tools:

PowerShell module (azure.datafactory.tools)

Azure DevOps extension (3 tasks)

Build → **Publish** → **Test**

https://sqlplayer.net/adftools/

DEMO #2 Deployment with PowerShell

SQL Player

Azure DevOps: Custom Tasks

Key concept:

- Free & Open-Source

- One task for everything when it comes to the publishing of ADF

- Basically, it is another (UI) layer on top of „azure.datafactory.tools" module

- Public Release (GA) since 23/12/2020

- Extension contains 3 tasks that cover full deployment life-cycle for ADF
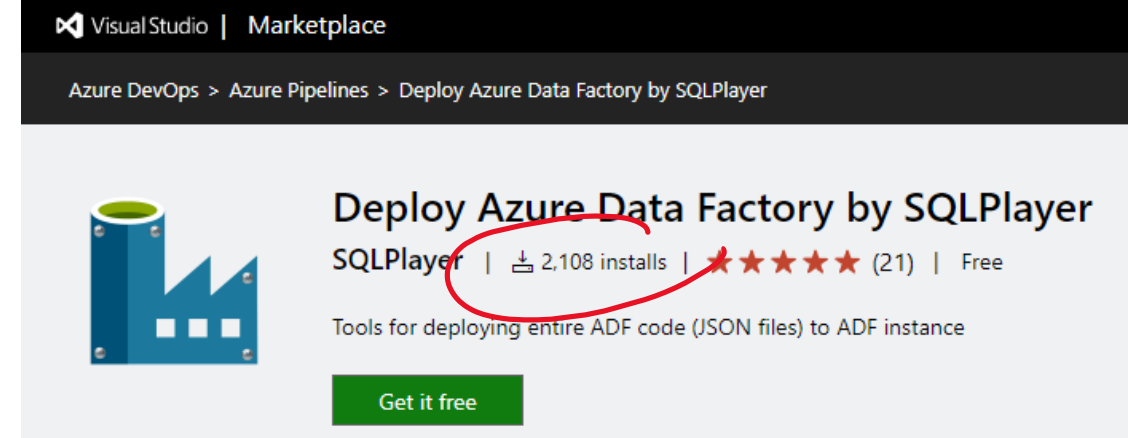
Any ideas or questions? Leave it here:
https://github.com/SQLPlayer/azure.datafactory.tools/issues

PowerShell module: azure.datafactory.tools

- Fully written in PowerShell, compatible with 5.1

- Uses Microsoft's PS module (Az.DataFactory) for management of ADF objects

- Available in [PowerShell gallery](#)

- `Publish-AdfV2FromJson` function - capabilities:

  - Creation of Azure Data Factory, if not exist (option)
  - Deployment of all type of objects: pipelines, datasets, linked services, data flows & power query, triggers, integration runtimes
  - Copes with dependencies (multiple levels) between objects when deploying (no more worrying about object names)
  - Build-in mechanism to replace the properties with the indicated values (CSV file)
  - Stop/start triggers (option)
  - Dropping objects when not exist in the source (code) (option)
  - Selective deployment: Filtering (include or exclude) objects to be deployed by name and/or type
  - Publish options allow you to control:
    - ▹ Whether stop and restarting triggers
    - ▹ Whether delete or not objects not in the source
    - ▹ Whether create or not a new instance of ADF if it not exist

- Free & Open-Source

https://github.com/SQLPlayer/azure.datafactory.tools

Task in Azure DevOps: Publish ADF (CD)

Publish Azure Data Factory
Deploys entire ADF (v2) from JSON files to ADF instance (adftools)

Key capabilities:

- Creation of Azure Data Factory, if not exist (option)
- Deployment of all type of objects: pipelines, datasets, linked services, data flows, triggers, integration runtimes, Managed Virtual Network, Managed Private Endpoint
- Copes with dependencies (multiple levels) between objects when deploying (no more worrying about object names)
- Build-in mechanism to replace the properties with the indicated values (CSV & JSON file format)
- Update, add or remove any property of ADF artefact
- Selective deployment declared in-line or by pointed file
- Stop/start triggers (option)
- Dropping objects when not exist in the source (code) (option)
- Filtering (include or exclude) objects to be deployed by name and/or type and/or type
- Filtering supports wildcards
- Publish options allow you to control:
  - Whether stop and restarting triggers
  - Whether delete or not objects not in the source
  - Whether create or not a new instance of ADF if it not exist
- Tokenisation in config file allows replace any value by Environment Variable or Variable from DevOps Pipeline
- Global Parameters
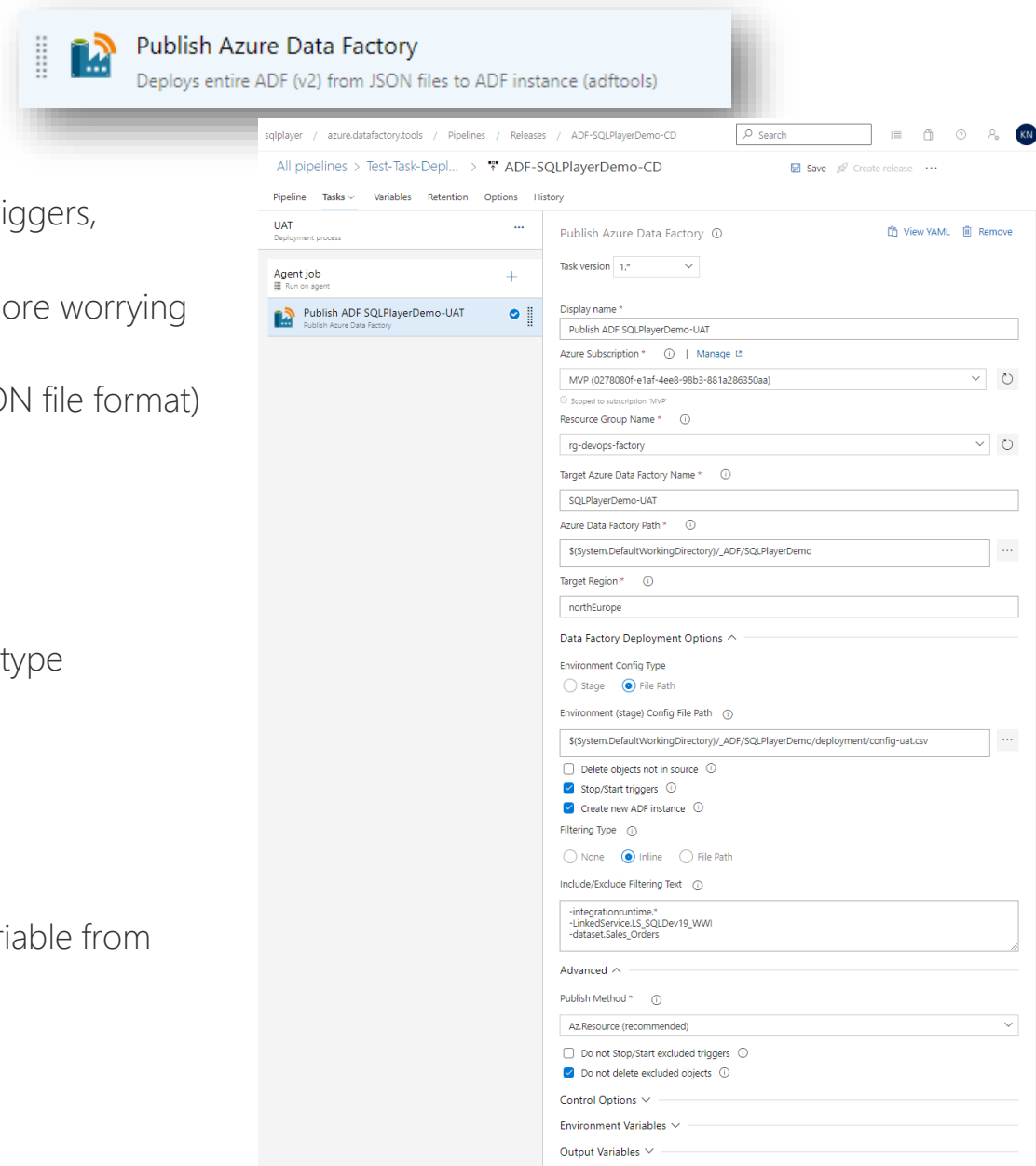
Selective deployment with PowerShell module

You can select objects by objects types & name using include or exclude option.

Allows to select the objects by belonging to a folder (picture)

Name can be **wildcarded**, so all such variants are possible:

```
trigger.*
dataset.DS_*
*.PL_*
linkedService.???KeyVault*
pipeline.ScdType[123]
*.*@testFolder
managedVirtualNetwork*.*
*managedPrivateEndpoint.*
```
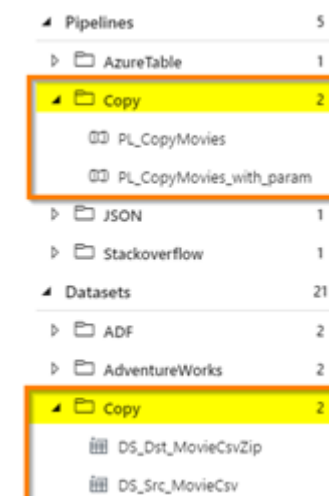
Selective deployment in Azure DevOps

☺ In <u>Azure DevOps Task</u> – the list can be provided either as (inline) **text** or from **file** in repo.

To simplify user experience – only one field is exposed in order to define include/exclude rules.

Therefore, an extra character should be provided before the name/pattern:

+ (plus) – for objects you want to include to a deployment

- (minus) - for objects you want to exclude from a deployment

If char is not provided – an inclusion rule would be applied.

Filtering Type ⓘ

○ None ⦿ Inline ○ File Path

Include/Exclude Filtering Text ⓘ

```
-integrationruntime.*
-LinkedService.LS_SQLDev19_WWI
-dataset.Sales_Orders
```

DEMO #3
Deployment with
Azure DevOps

PowerShell module: Parameters for Stages

How are parameters passed into the deployment? Config CSV File:

```
 1  type,name,path,value
 2  linkedService,LS_AzureKeyVault,typeProperties.baseUrl,"https://kv-blog-uat.vault.azure.net/"
 3  # This is comment - the line will be omitted
 4  linkedService,LS_BlobSqlPlayer,typeProperties.connectionString,"DefaultEndpointsProtocol=https;Acco
 5  pipeline,PL_CopyMovies,activities[0].outputs[0].parameters.BlobContainer,UAT
 6  pipeline,PL_CopyMovies_with_param,parameters.DstBlobContainer.defaultValue,"$($Env:Environment)"
 7  pipeline,PL_Wait_Dynamic,parameters.WaitInSec,"{'type': 'int32','defaultValue': 22}"
 8  # MINUS means the desired action is to REMOVE encryptedCredential:
 9  linkedService,BlobSampleData,-typeProperties.encryptedCredential,
10  # PLUS means the desired action is to ADD new property with associated value:
11  linkedService,BlobSampleData,+typeProperties.accountKey,"$($Env:VARIABLE)"
```

Option 1:

Any variables can come from DevOps Pipeline, either normal as well as sensitive values.

To apply replacement for secret values:

Environment Variables must be mapped (Microsoft recommendation).

Option 2:

Another option would be reading secrets directly from provided Azure Key Vault.

Therefore, "Replacement" task is still recommended here as an alternative as for now.

Bonus capabilities

SQL Player
Play with data & have fun!

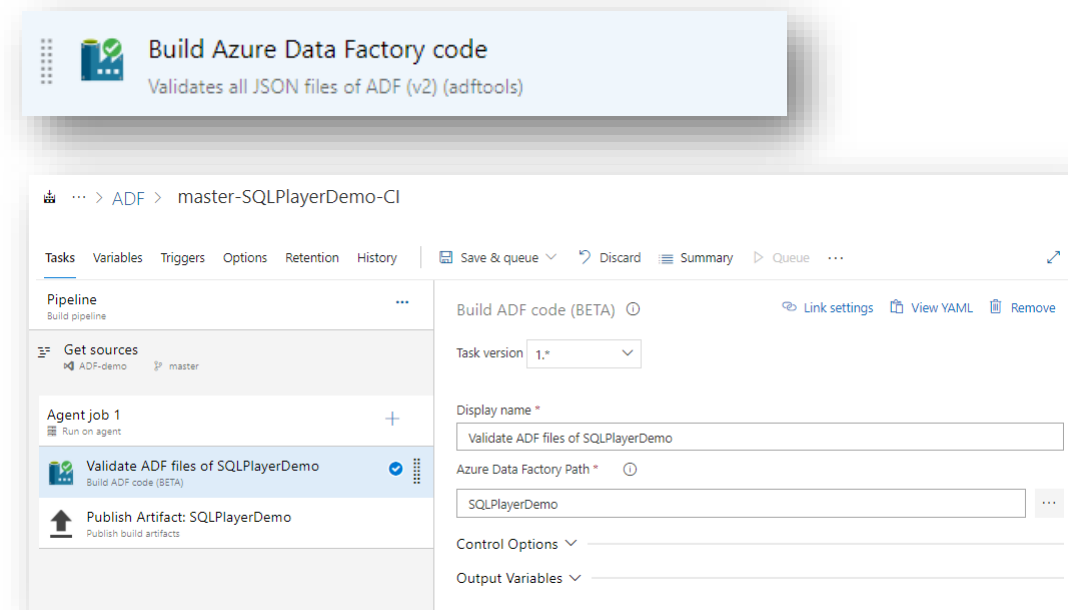# New task in Azure DevOps: Build ADF (CI)

The task has 2 modes:

- **Build only**
  - Reads all files and validates its json format
  - Checks whether all dependant objects exist
  - Checks whether file name equals object name

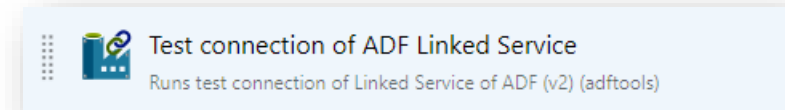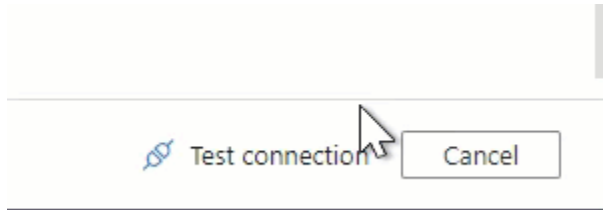- **Validate & Export ARM Template**
  - uses ADFUtilities NPM package provided by Microsoft
  - Counterpart of Validate all and Export ARM Template in ADF UI

New task in Azure DevOps: Test Connection

- Run „Test connection" for a Linked Service

- Smoke tests of (some) Linked Services

- In preview

Summary

SQL Player
Play with data & have fun!

Takeaway: ADF deployment - Possible paths

SQL Player
Play with data & have fun!

adf_publish branch

main / master branch

Azure DevOps

Export

Pull

Deprecated

Export

Validate

ARM Template

Code artifact

Build (CI) Pipeline

Stop Triggers

Deploy ARM Template

Start Triggers

Deploy + [Test]

Release (CD) Pipeline

Azure

Stage

Dev

Test

Prod

TOOLS

#adftools

npm library

#Adftools - Resources

- https://sqlplayer.net/adftools

- https://github.com/SQLPlayer/azure.datafactory.tools

  - Issues

  - Discussions, FAQ

- PowerShell Gallery: azure.datafactory.tools

- Marketplace: Deploy Azure Data Factory (extension for Azure DevOps)

- Microsoft ADF (npm) Utilities

- https://github.com/SQLPlayer/azure.**synapse**.tools (preview!)

Questions?
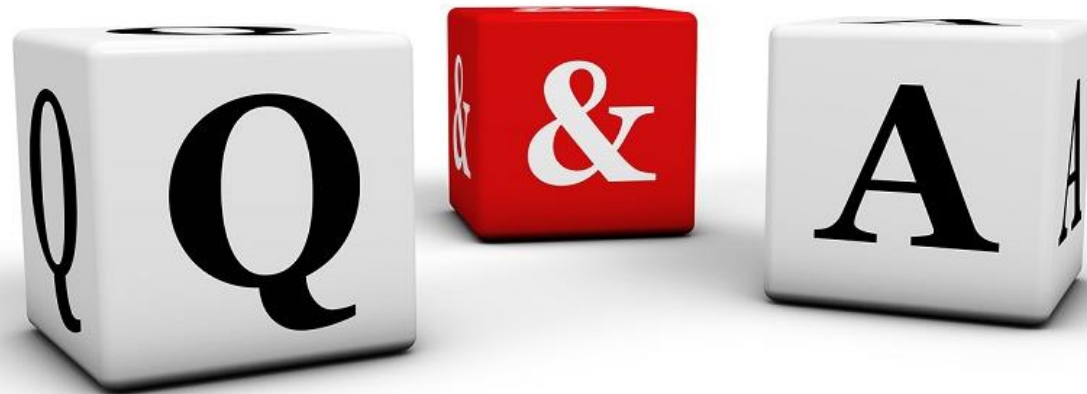
Thank you!

✉️ kamil@sqlplayer.net

🐦 @NowinskiK   @SQLPlayer

🔗 SQLPlayer.net

⬛ https://github.com/NowinskiK/CommunityEvents

Kamil Nowinski
Microsoft Data Platform MVP
Analytics Architect, Azure DevOps Engineer Expert