Bob Duffy, MCA, MCM, MVP

PowerBI DevOps for Enterprise Projects

Bob Duffy

- 30 years in database sector, 250+ projects
- SQL Server MCA, MCM, MVP
- SSAS Maestro
- Senior Data Platform Consultant with Microsoft 2005-2008
- Database Architect at Prodata SQL Centre of Excellence

https://prodata.ie/blogs/
bob@prodata.ie

@bob_duffy

Agenda

- Source Code 101 - Why Use GIT ?
- A Sample Deployment Environment
- Deployment Techniques
  - Method 1 - Use and App/workspace
  - Method 2 – PowerBI Deployment Pipeline
  - Method 3 – Powershell/REST
  - Method 4 – Azure DevOps Pipelines
- Refreshing via API and Monitoring

Why use GIT to store PBIX?

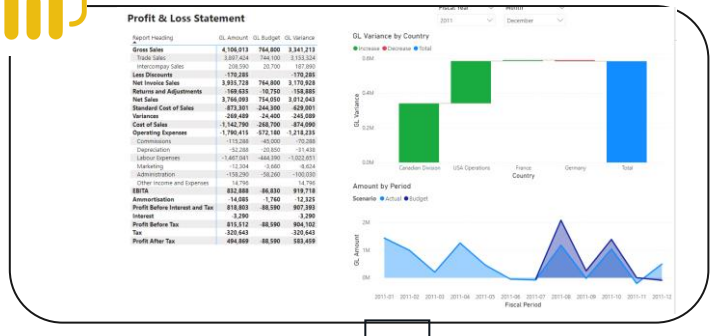
- Good Points
  - Allows for backup versioning and rollback
  - Check In/Out for developers
  - Can use branching for storing main v feature versions
  - Requirement for official Deployment Pipelines
    (Microsoft uses it so why shouldn't we)
- Bad Points
  - Learning GIT
  - Large objects in GIT "blows" up size
    (do not mix PBIX in a general purpose GIT repos!)
  - Some recommendations for one drive.
  - No Source Code integration into IDE (yet)
  - GIT Merge doesn't really work. Can loose work.

GIT Merge - YouTube

Demo

The Deployment Process

AdventureWorks [DEV]



\Main

\Feature

Cloud Gateway

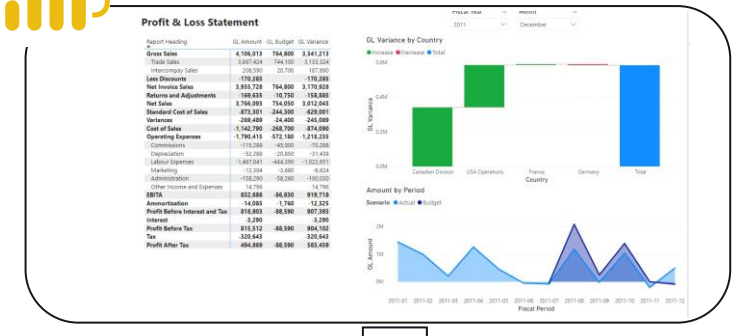
AdventureWorksDW

AdventureWorks [Test]

On Premise Gateway

AdventureWorks2017

AdventureWorks

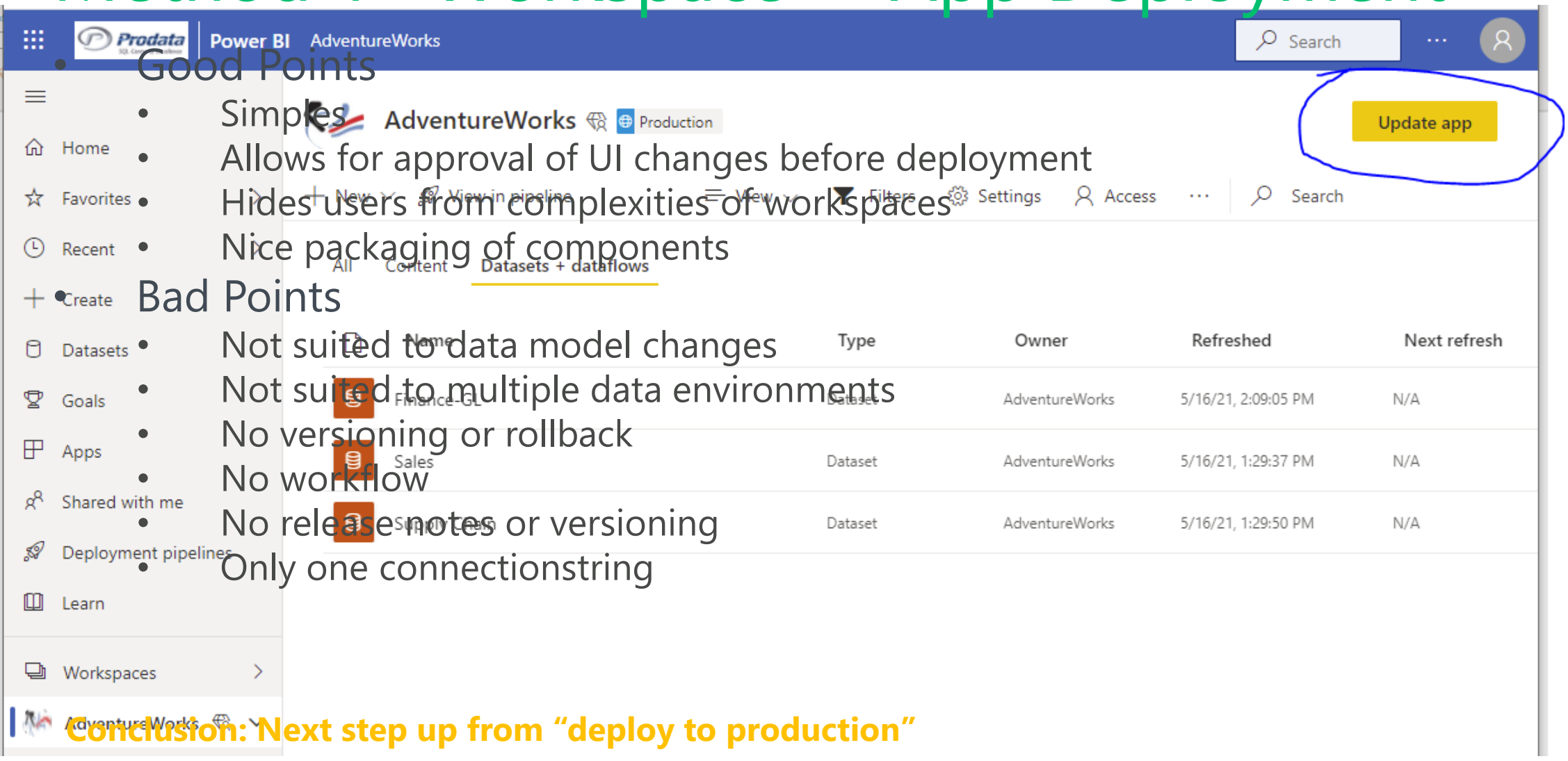


Cloud Gateway

SQLDW

Method 1 –Workspace=>App Deployment

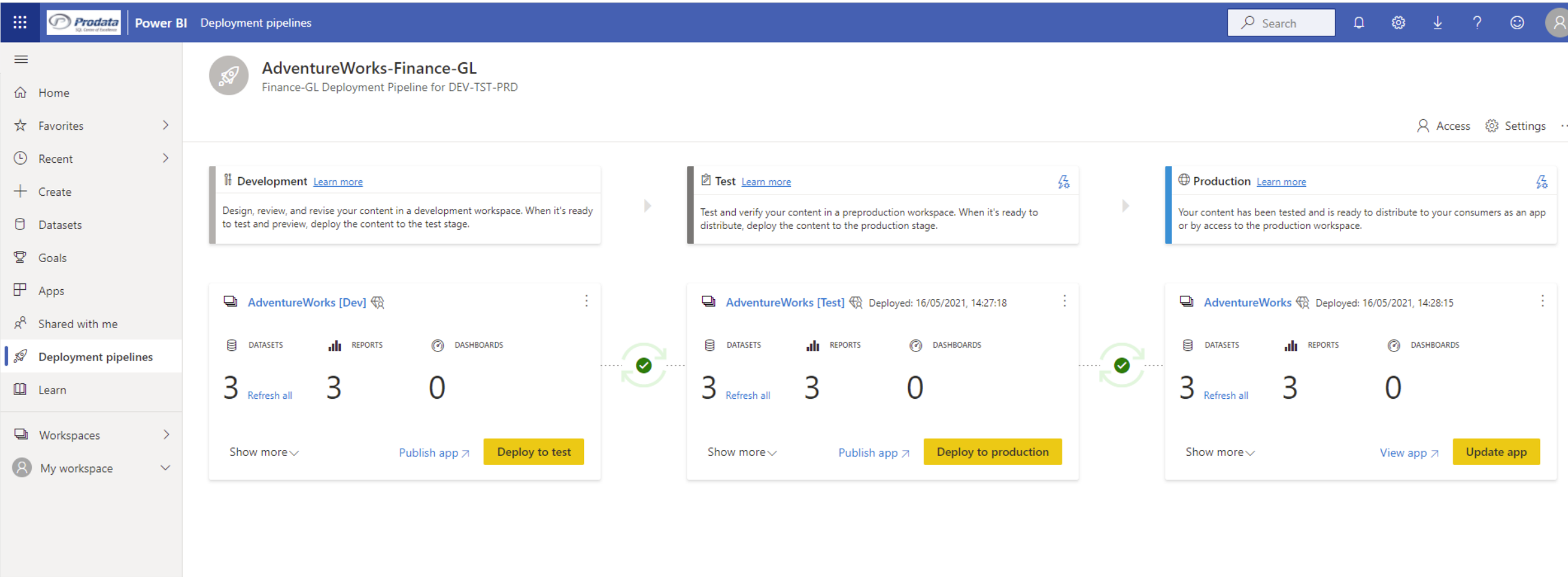
- Good Points
  - Simples
  - Allows for approval of UI changes before deployment
  - Hides users from complexities of workspaces
  - Nice packaging of components
- Bad Points
  - Not suited to data model changes
  - Not suited to multiple data environments
  - No versioning or rollback
  - No workflow
  - No release notes or versioning
  - Only one connectionstring



Conclusion: Next step up from "deploy to production"

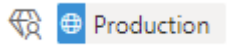
Demo

Method 2 – Deployment Pipelines



Demo

Method 2 – Deployment Pipelines

- Good Points
  - Can compare different environments
  - Super Easy and intuitive
  - Handles some connection string changes
  - I love the little tag on workspaces 

- Bad Points
  - No Rollback or release numbering or notes
  - Makes assumptions on dev connection strings
  - Not GIT Integrated. Need to manually upload to dev Workspace
  - Doesn't bind gateways
  - Doesn't deal with linked reports
  - Premium Only ($$$)
  - Its called pipelines but its not a proper azure DevOps pipeline. Why not ;-((

Conclusion: is a V1 effort. useful for some people. Not "Enterprise" enough for my projects

No news on V2 AFAIK, but this could be the start of something great

Demo

Method 3 – Powershell and REST

02 Deploy-Dev-TST-PRD.ps1 > ...

```powershell
# Sample Script to automate publishing of reports and DataSets using PowerBI REST and CmdLets
# Install-Module MicrosoftPowerBIMgmt
# Install-Module pbiTools

# https://docs.microsoft.com/en-us/powershell/module/microsoftpowerbimgmt.data/?view=powerbi-ps

$ds = Connect-PowerBIServiceAccount
$UserName =$ds.UserName

#Parameters
$Path= "C:\Projects\Prodata\AdventureWorks\repos\PowerBI\Finance"
$SqlServer="dev-sql.roebuck.local"
$WorkspaceName ="Adventureworks [Dev]"
$DataSetName ="Finance-GL"
$SqlDatabaseName="AdventureWorksDW"

#Deploy to DEV, change adata source and map Gateway
New-pbiReport -path $Path -Workspace $WorkSpaceName -UserName $UserName -SqlServer $SqlServer

#Refresh DEV (if needed)
Invoke-pbiRefresh -Workspace $WorkspaceName  -Dataset $DataSetName -Wait  -SqlServer "localhost" -SqlDatabaseName $SqlDatabaseName -SqlSchemaName "audit"  -SqlTabl

#Deploy to DEV, change adata source and map Gateway
$WorkspaceName ="Adventureworks [Test]"
$SqlServer="test-sql.roebuck.local"
New-pbiReport -path $Path -Workspace $WorkSpaceName -UserName $UserName -SqlServer $SqlServer

#Deploy to PRD
$WorkspaceName ="Adventureworks"
$SqlServer="prd-sql.roebuck.local"
New-pbiReport -path $Path -Workspace $WorkSpaceName -UserName $UserName -SqlServer $SqlServer
```

Demo

Method 3 – Powershell and REST

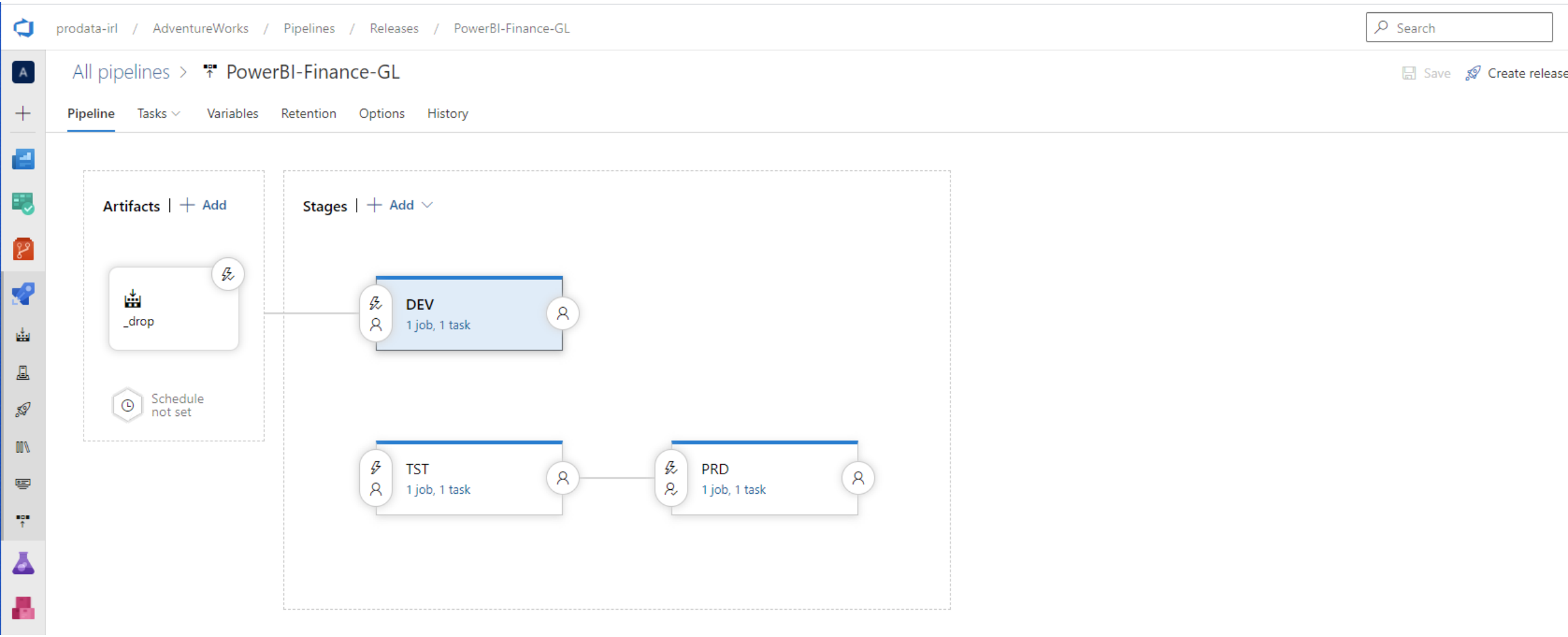
- Good Points
  - Easy to call, albeit some Powershell
  - Great for event driven refresh (we use this)
  - Easy to schedule if needed
  - Extensible for complexities (gateway and linked reports)

- Bad Points
  - No rollback
  - No deployment library
  - No schema compare or release numbering
  - Mostly from collaborative branch – drift danger.

Sample Powershell cmdLets

- Connect-PowerBIServiceAccount
- Get-PowerBIWorkspace
- Get-PowerBIDataset
- Get-PowerBIReport
- New-PowerBIReport
- Invoke-PowerBIRestMethod
  - groups/$WorkspaceId/datasets/$DataSetId/DiscoverGateways
  - groups/$WorkspaceId/datasets/$DataSetId/BindToGateway
  - groups/$WorkspaceId/datasets/$DataSetId/datasources
  - groups/$WorkspaceId/datasets/$DataSetId/updatedatasources
  - groups/$WorkspaceId/datasets/$DataSetId/refreshes [POST]
  - groups/$WorkspaceId/datasets/$ DataSetId/refreshes?$top=1 [GET]
  -

Method 4 – Full Azure Pipelines



Demo

Method 4 – Full Azure Pipelines

- Good Points
  - Most powerful and extensible pipeline
  - Can integrate some testing
  - Same pipelines as used by other solution components.
  - Great versioning, rollback and SDLC Integration
- Bad Points
  - More difficult to learn than PowerBI Pipelines
  - Needs DevOps License
  - Still has “bloat” of PBIX inside repos
  - No compare as such
  - Can be not granular enough

Demo

After Deployment- Refresh and Monitor

- Default Refresh is time based. Doesn't work for many projects
- REST API Allows for even driven refresh
- With some Powershell can log to SQL

Demo

So what did we learn ?

- Many ways of doing deployments
- Don’t overcook based on your project size
- Use Workspace=> App as first shirt size
- PowerBI pipelines for premium projects
- Powershell or Azure pipelines for Pro projects
- Azure Pipelines for complex or full stack projects

- Use Powershell to get the status of PBIX’s and monitor
- Use Powershell/REST to go event driven on refresh

Demo

Thanks. Q +A and Links

- Power BI Deployment Pipelines Best Practices
https://docs.microsoft.com/en-us/power-bi/create-reports/deployment-pipelines-best-practices

- Power BI REST APIs
https://docs.microsoft.com/en-us/rest/api/power-bi/

- Power BI cmdlets
https://docs.microsoft.com/en-us/powershell/power-bi/overview?view=powerbi-ps

- pbiTools – Sample Powershell wrapper/Library
https://www.powershellgallery.com/packages/pbiTools/1.4

- Azure Pipelines
https://docs.microsoft.com/en-us/azure/devops/pipelines/?view=azure-devops

My wish List

- GIT Integration !
- Separate data from model/reports
  (and exclude from source control)
- Create proper Azure Pipeline, not Power BI Pipelines
- Better Handling for gateways and linked reports
- Finish the Powershell cmdLets
- Expose App Publish as API

Demo