



>>>>>>

dev>scope

Rui Romano

+ 15 years "playing" with data

Head of BI Team @ **DevScope** – OPorto, **Portugal**

#DataArchitect #DataEngineer #PowerBIAddict
#MSFTBI #DataPlatformMVP #PBIPortugal

 Rui.Romano@DevScope.net

 [@RuiRomano](https://twitter.com/RuiRomano)

 <https://www.linkedin.com/in/ruiromano/>

 <https://ruiromanoblog.wordpress.com>

 <https://www.meetup.com/Power-BI-Portugal>



dev>scope





Power BI Monitoring 101

- Not a Power BI Intro!
- Why you need monitoring?
- Report Demo
- How to do it? – Fast Paced 😊

Focus on the possibilities not the how...

Can you answer these questions?

- Who are most active users?
- How many distinct users? Per Month? Per day? Per hour?
- Which Reports/Workspaces/Datasets/Apps are mostly used?
- Are Personal Workspaces being used? How is content being shared?
- Do your users access from Browser/Mobile/Excel? Which browser?
- Top used DataSource's? FileSystem/OneDrive? SQL? Sharepoint? DataLake?
- DataSets Refreshing trends/average/errors?
- DataSets/Reports not used in more than 1 year?
- How many users with license don't use Power BI in more than 3 months?

NO?

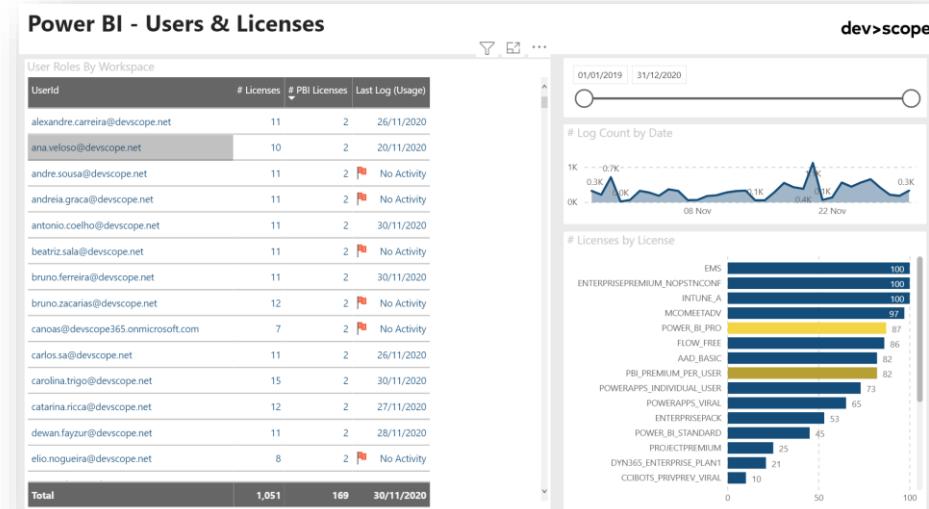
You are “driving blind” and certainly, a **BIG Reality Check** is ahead of you...

Monitoring is one of the main pillars of a good Power BI Governance strategy

This session goal is to motivate you to look at this data, it's always important to do the “BI of the BI”!



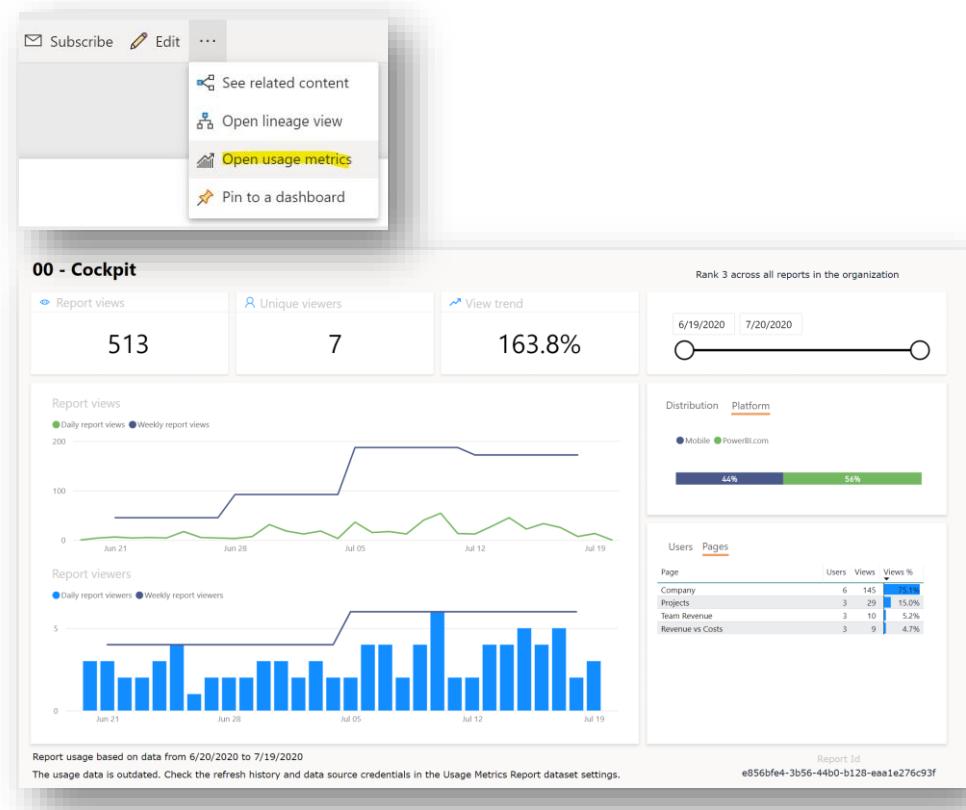
DEMO – Power BI Monitor



What is available Out of the Box?

Power BI Report Usage

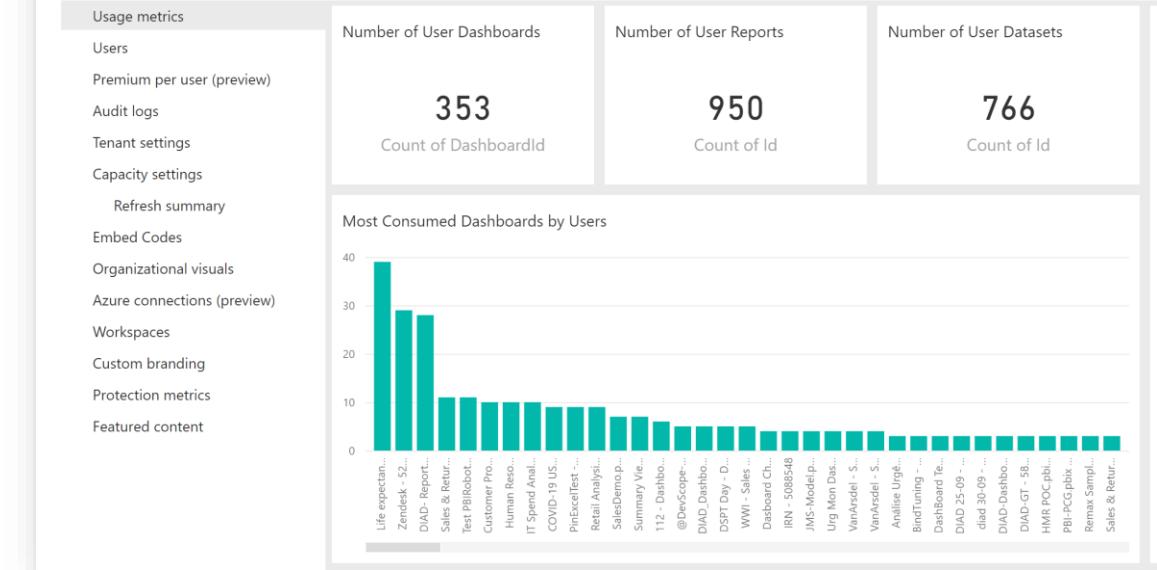
- Workspace level
- Page metrics
- Client Telemetry data (time to open report)
- 7 days



Usage Metrics in Admin Portal

- Tenant Level
- Zero Interactivity and Customization
- No Refresh Control

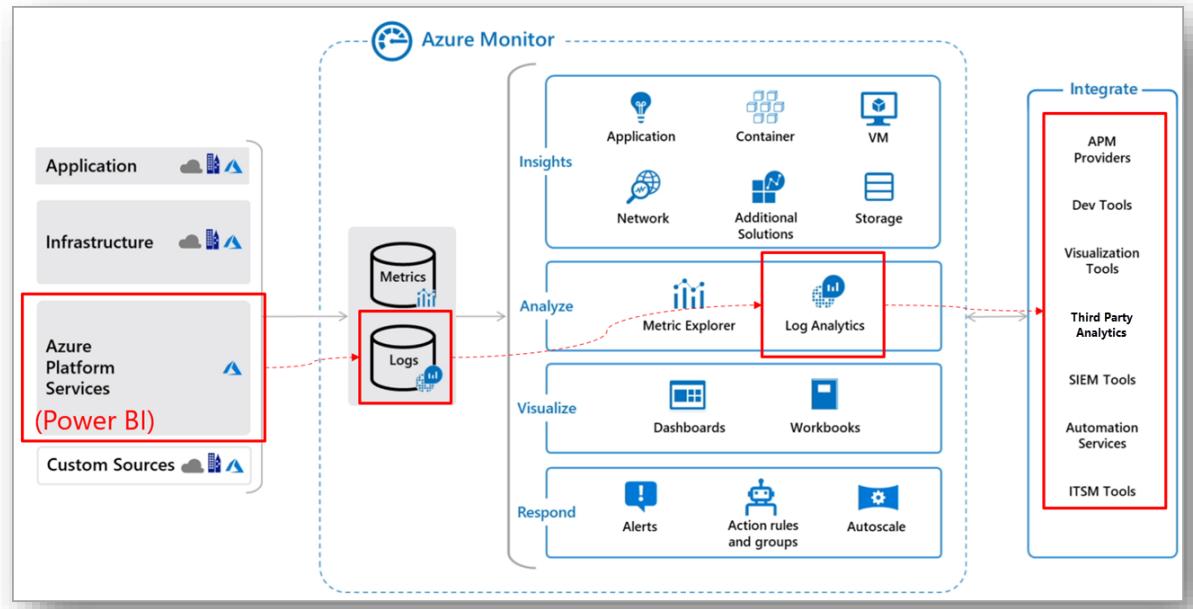
Admin portal



What is available Out of the Box?

Power BI & Log Analytics

- Dataset level – Queries, Refresh Commands,...
 - Similar to [Azure AS Diagnostics](#)
 - Very useful to track performance issues on a large tenant
 - Ex: Correlating a usage spike to the DAX queries
 - Pre-Built [Power BI Report Template](#)
 - **Preview Limitations**
 - Cannot connect multiple workspaces to same Log Analytics



The screenshot shows the Azure Settings page for an admin user. The top navigation bar includes 'About', 'Premium', and 'Azure connections (preview)'. Below the navigation, there are two main sections: 'Storage' and 'Log Analytics'. The 'Log Analytics' section is expanded, showing a connection to a Log Analytics workspace named 'admir' (resource group 'rg-byola', workspace ID 'a1...7e'). The workspace was configured by Admin Admin on 2020-10-12T12:00:00Z. A large red 'Disconnect from Azure' button is prominently displayed at the bottom of this section.

Settings

admin [REDACTED]

About Premium Azure connections (preview)

▶ Storage

◀ Log Analytics

Connect an Azure Log Analytics workspace to collect usage and performance logs for this workspace. [Learn more](#)

Subscription a1 [REDACTED] 7e

Resource group rg-byola

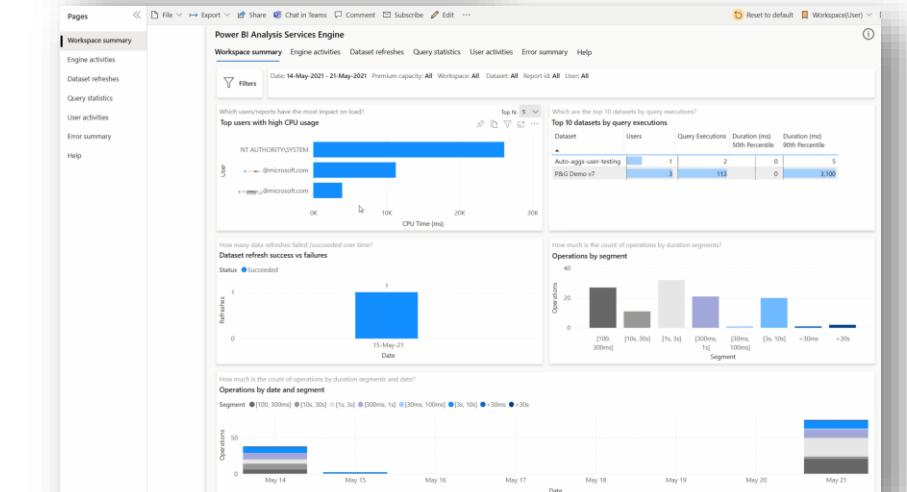
Log Analytics workspace admir

Configured by Admin Admin on 2020-10-12T12:00:00Z

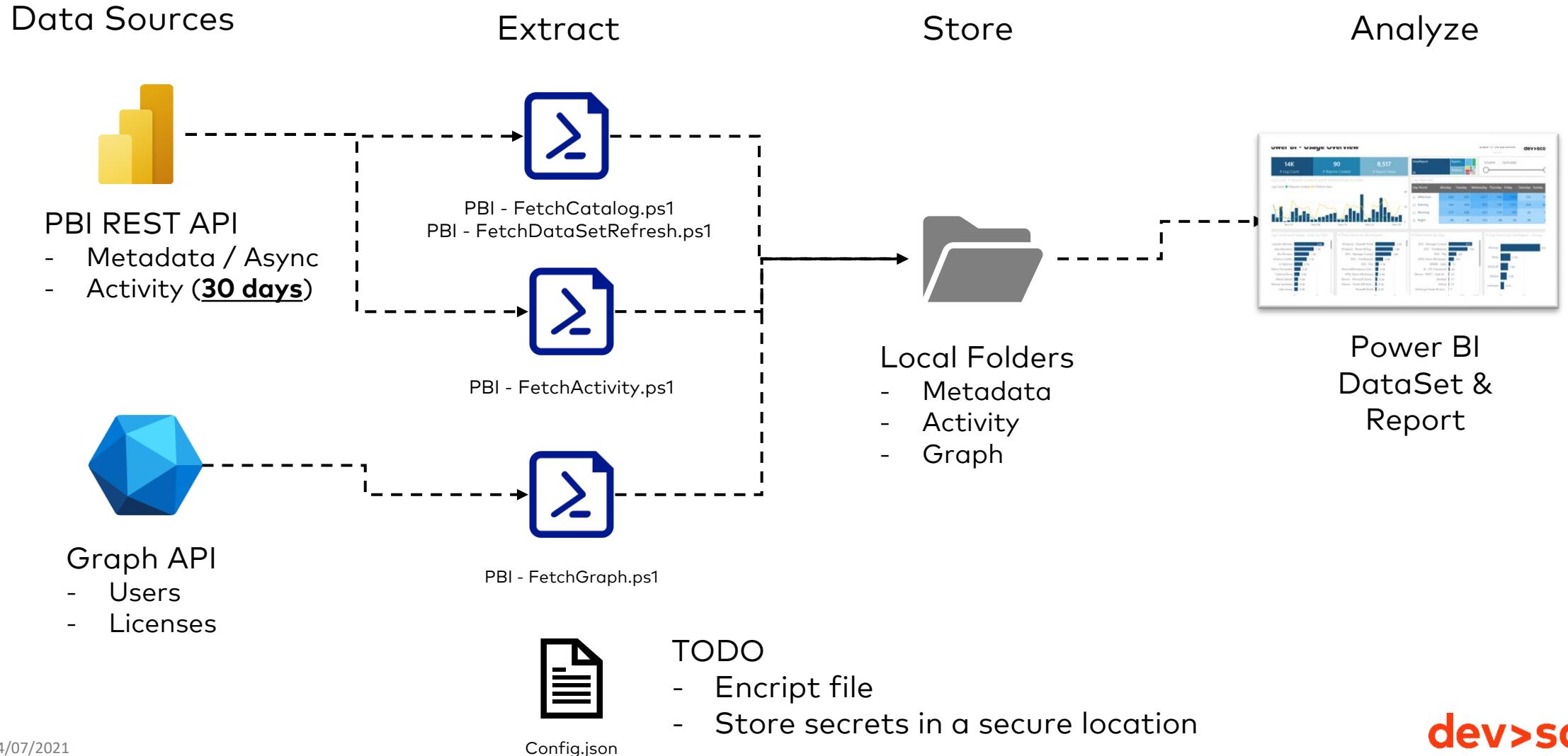
Disconnect from Azure

Disconnecting will stop usage and performance data from flowing into your Azure Log Analytics workspace.

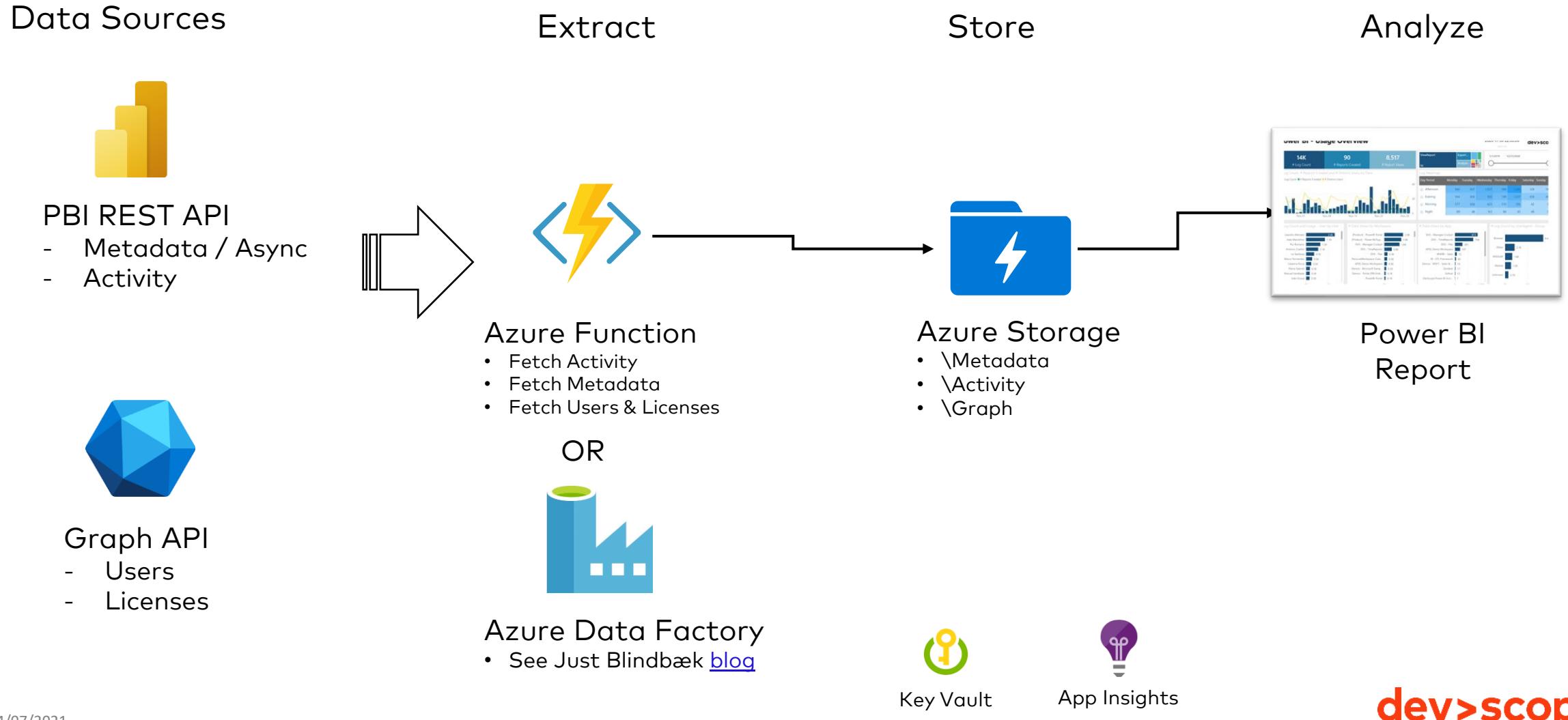
Disconnect from Azure



Architecture – Simplest and Easy to Share



Architecture – Recommended (one of many possibilities)



API's Overview

Scope	Resource	API
Power BI Metadata	Workspaces DataSets Reports Dashboards	Power BI Admin Scan APIs
	Users	Power BI Admin API – GetGroupsAsAdmin + Expand Users
	RefreshHistory	Power BI Admin API – GetGroupsAsAdmin + Expand DataSets Power BI API - Get Refresh History In Group
Activity	Power BI Activity Logs	Power BI Admin API - ActivityEvents
Users & Licenses	Users & Licenses	Microsoft Graph API – Users
	Licenses Details	Microsoft Graph API – SubscribedSKUs

The screenshot shows the Microsoft Docs page for the Power BI REST APIs. The page title is "Power BI REST APIs" and it includes a brief description: "Power BI REST API provides service endpoints for embedding, administration, and user resources." The page features a sidebar with a navigation tree and a main content area with a table titled "REST Operation groups".

Operation group	Description
Admin	Operations for working with administrative tasks.
Apps	Operations for working with Apps.
Available Features	Operations that return available features.
Capacities	Operations for working with capacities.
Dashboards	Operations for working with dashboards.
Dataflows	Operations for working with dataflows.
Datasets	Operations for working with datasets.
Embed Token	Operations for working with embed tokens.
Gateways	Operations for working with gateways.
Groups	Operations for working with groups.
Imports	Operations for working with imports.
Pipelines	Operations for working with deployment pipelines.
Push Datasets	Operations for working with push datasets.
Reports	Operations for working with reports.
Template Apps	Operations for working with Template Apps.
Users	Operations for working with users.

Requirements

- Be a Power BI Administrator (or friend to one 😊)
- Permissions to create an Azure AD Application / Service Principal
- Permission to create an Azure AD Security Group

Directory roles

↑ Sort

To assign custom roles to a user, your organization needs Azure AD Premium P1 or P2.

Choose admin roles that you want to assign to this user. [Learn more](#)

<input type="checkbox"/>  Power BI administrator	Can manage all aspects of the Power BI product.
---	---

1 Rui Romano AD (MVP Subscription) | User settings 2

Azure Active Directory

Overview

Preview features

Diagnose and solve problems

Manage

Users

Groups

External Identities

Save Discard

Enterprise applications

Manage how end users launch and view their applications

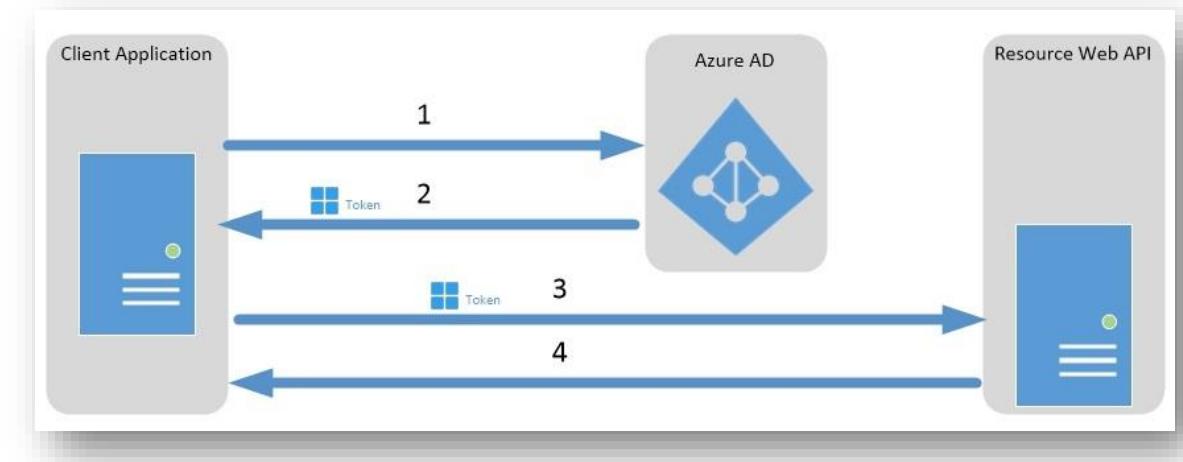
App registrations

Users can register applications ⓘ

Yes No

API Authentication

- OAuth 2.0
- Possible Authentication Flows:

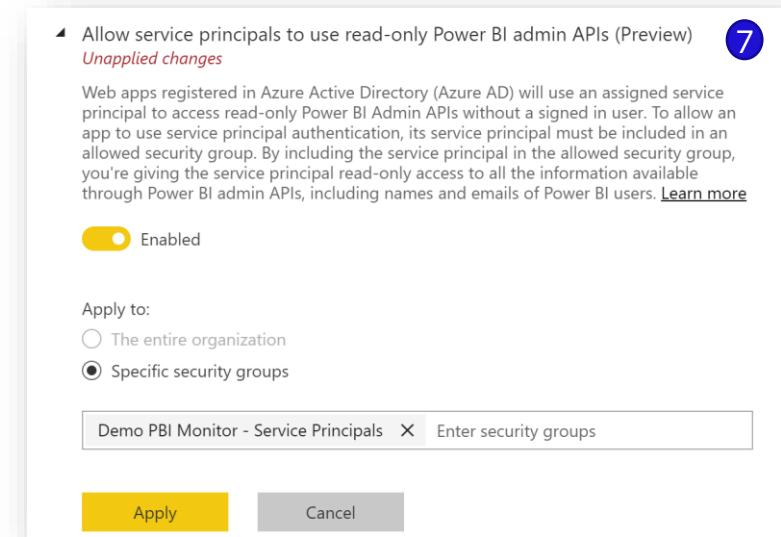


Authentication Flow	Requirements
<u>Client Credentials Flow</u>	Azure AD Service Principal Permissions needed: <ul style="list-style-type: none"> • Power BI Admin Authorization (read only) • Admin Permissions to read Graph API<ul style="list-style-type: none"> • User.ReadAll Organization.ReadAll
<u>Device Code Flow</u>	Power BI Administrator Account
Username & Password	Power BI Administrator Account
Not Recommended	

Service Principal Step by Step

1. Go to [Azure AD Active Directory](#)
2. Go to [App Registrations](#) and create a new App (leave defaults)
3. Generate a new App Secret
4. Save the [App Id](#), [App Secret](#) & [Tenant Id](#)
5. [Create an Azure AD Security Group](#)
6. Add the Service Principal to the Security Group as a member
7. Authorize the Security Group in Power BI Admin Portal
8. **Optional** - Authorize the Service Principal to Access Graph API

Note: You don't need to add any Power BI API Permissions



+ Add a permission ✓ Grant admin consent for Rui Romano AD (MVP Subscription) 8

API / Permissions name	Type	Description
Microsoft Graph (3)		
Organization.Read.All	Application	Read organization information
User.Read	Delegated	Sign in and read user profile
User.Read.All	Application	Read all users' full profiles



milestones

- ✓ IT/BI Admin Support
- ❑ Extract Data
- ❑ Data Store
- ❑ Power BI DataSet
- ❑ Power BI Report

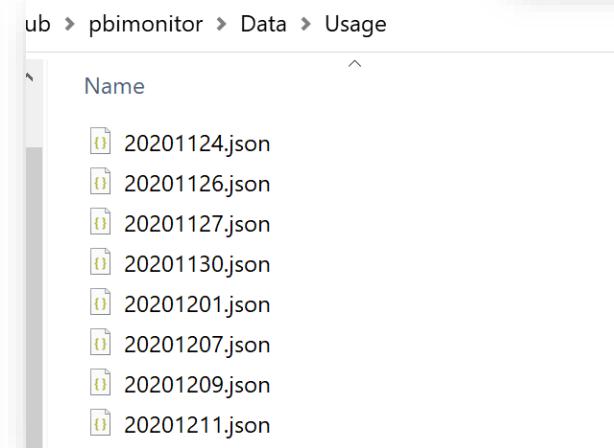
Script PBI - FetchActivity.ps1

- Incrementally fetch Power BI Activity
Uses [PowerBIPS](#) module
- Uses the Power BI Admin API:
[Get Activity Events](#)
- Can only go back **30 days**

```
1  {
2      "ServicePrincipal": {
3          "AppId": "",
4          "AppSecret": "",
5          "TenantId": ""
6      },
7      "Catalog": {
8          "LastRun": "2021-02-18T00:00:00.000000Z"
9      },
10     "Activity": {
11         "LastRun": "2021-02-18T00:00:00.000000Z"
12     }
}
```

```
30
31     if ($config.Activity.LastRun)
32     {
33         $pivotDate = [datetime]::Parse($config.Activity.LastRun)
34     }
35     else
36     {
37         $config | Add-Member -NotePropertyName "Activity" -NotePropertyValue @{"LastRun" = $null}
38         $pivotDate = [datetime]::UtcNow.Date.AddDays(-30)
39     }
40
41     # Gets audit data daily
42
43     while($pivotDate -le [datetime]::UtcNow)
44     {
45         Write-Host "Getting audit data for: '$($pivotDate.ToString('yyyyMMdd'))'"
46         $odataParams = "startDateTime='$(($pivotDate.ToString('s')))&endDateTime='$(($pivotDate.AddHours(1)).ToString('s'))'"
47         $audits = @(Invoke-PBIREquest -authToken $authToken -resource "activityevents" -admin -odata $odataParams)
48
49         if ($audits.Count -gt 0)
50         {
51             Write-Host "'$($audits.Count)' audits"
52             $outputFilePath = "$outputPath\$($pivotDate.ToString('yyyyMMdd')).json"
53             ConvertTo-Json $audits | Out-File $outputFilePath -force
54         }
55         else
56         {
57             Write-Warning "No audit logs for date: '$($pivotDate.ToString('yyyyMMdd'))'"
58         }
59
60         $config.Activity.LastRun = [datetime]::UtcNow.Date.ToString("o")
61
62         $pivotDate = $pivotDate.AddDays(1)
63
64         # Save config
65         ConvertTo-Json $config | Out-File $configPath -force
66     }
67 }
```

```
PS C:\@Repos\Github\pbimonitor> C:\@Repos\Github\pbimonitor>
Getting OAuth Token
Getting audit data for: '20210218'
'1322' audits
Getting audit data for: '20210219'
'223' audits
PS C:\@Repos\Github\pbimonitor>
```



Script – PBI - FetchCatalog.ps1

- Snapshot the entire tenant metadata:

- Workspaces (personal included)
- DataSets
- DataSources
- Reports
- Dashboards
- Users

- Uses the new [Async API](#)

Faster & Incremental
Lineage + Schema ([preview](#))

- Missing on Async API:

Workspace Users + Roles

- [Admin Get Groups](#) + \$expand=users
- Admin Apps

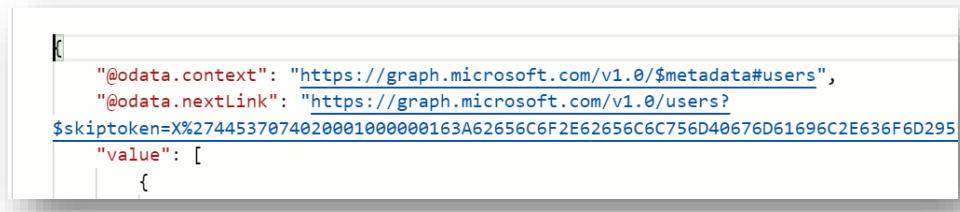
```
do
{
    try
    {
        $workspacesBatch = @($workspacesModified | Select -First $batchCount -skip $skip)
        if ($workspacesBatch)
        {
            Write-Host "Requesting workspace scan: $($skip + $batchCount) / $($workspacesModified.Count)"
            $bodyStr = @{"workspaces" = $workspacesBatch.Id} | ConvertTo-Json
            $workspacesScanRequests += Invoke-PBIRequest -authToken $authToken -resource "workspace"
            $skip += $batchCount
        }
    }
    catch [System.Net.WebException]
    {
        $ex = $_.Exception
        $statusCode = $ex.Response.StatusCode
        if ($statusCode -eq 429)
        {
            $waitSeconds = [int]::Parse($ex.Response.Headers["Retry-After"])
            Write-Host "429 Throttling Error - Need to wait $waitSeconds seconds..."
            Start-Sleep -Seconds ($waitSeconds + 5)
            $authToken = Get-PBIAuthToken -clientId $config.ServicePrincipal.AppId -clientSecret $config.ServicePrincipal.Password
        }
    }
}
while($workspacesBatch.Count -ne 0 -and $batchCount -lt 5000)
```

```
PS C:\@Repos\github\pbimonitor> C:\@Repos\Github\pbimonitor>
Fetching 5000 /admin/workspaces
Getting workspaces to scan
Since: 2021-02-18T00:00:00.0000000z
Modified workspaces: 5
Requesting workspace scan: 100 / 5
Waiting for scan results...
Scan 'bbf36328-5210-4006-bd45-830cf22e2793' : 'succeeded'
Scan Result 'bbf36328-5210-4006-bd45-830cf22e2793' : '5'
Elapsed: 9.0469164s
PS C:\@Repos\Github\pbimonitor>
```

Github > pbimonitor > Data.DVS > Catalog > 2021 > 02 > 17 >	
Name	Date modified
scans	17/02/2021 15:44
WORKSPACES.USERS.JSON	17/02/2021 15:16

Script – PBI - FetchGraph.ps1

- Snapshot tenant:
 - Users & Assigned Licenses
 - Tenant Subscribed SKUs
- Uses the Microsoft Graph API
 - Users
 - SubscribedSkus
- Paginated API



```

New-Item -ItemType Directory -Path $outputPath -ErrorAction SilentlyContinue | Out-Null
# Get the authentication token
$authToken = Get-AuthToken -resource "https://graph.microsoft.com" -appid $servicePrincipal.AppId -
$graphUrl = "https://graph.microsoft.com/beta"
Write-Host "Getting Users from Graph"
$users = Read-FromGraphAPI -accessToken $authToken -url "$graphUrl/users?$select=id,mail,companyName"
$filePath = "$outputPath\Graph.Users.json"
$users | ConvertTo-Json -Compress -Depth 5 | Out-File $filePath -Force
Write-Host "Getting SKUs from Graph"
$skus = Read-FromGraphAPI -accessToken $authToken -url "$graphUrl/subscribedskus?$select=id,capabilities"
$filePath = "$outputPath\Graph.SKUs.json"
$skus | ConvertTo-Json -Compress -Depth 5 | Out-File $filePath -Force
  
```

ithub > pbimonitor > Data.DVS > Graph > 2021 > 02 > 17

Name	Date modified
subscribedSkus.json	17/02/2021 18:50
users.json	17/02/2021 18:50

Script – PBI - FetchDataSetRefresh.ps1

- There is no Admin API to get DataSetRefresh History
- Ensure the service principal is a member of every workspace to monitor, manually or [script](#)
- Loop all datasets and call “Refreshes” Api that get the latest refreshes for the dataset

```

96 foreach($workspace in $workspaces)
97 {
98     $item++
99     write-Host "Processing workspace: '$($workspace.Name)' $item/$total"
100    write-Host "Datasets: $($workspace.datasets.Count)"
101
102    $refreshableDatasets = @($workspace.datasets | ? { $_.isRefreshable -eq $true -and $_.addRowsAPIEnabled -eq $false})
103    write-Host "Refreshable Datasets: $($refreshableDatasets.Count)"
104
105    foreach($dataset in $refreshableDatasets)
106    {
107        try
108        {
109            write-Host "Processing dataset: '$($dataset.name)'"
110            write-Host "Getting refresh history"
111
112            $dsRefreshHistory = Invoke-PBIRequest -authToken $authToken -resource "datasets/$($dataset.id)/refreshes" -groupId $workspace.id
113
114            if ($dsRefreshHistory)
115            {
116                $dsRefreshHistory = $dsRefreshHistory | Select *, @{Name="dataSetId"; Expression={ $dataset.id }}, @{Name="dataSet"; Expression={ $dataset.name }}, @{Name="group"; Expression={ $workspace.name }}, @{Name="configuredBy"; Expression={ $dataset.configuredBy }}, @{Name="lastRefreshTime"; Expression={ $dataset.lastRefreshTime }}, @{Name="lastRefreshStatus"; Expression={ $dataset.lastRefreshStatus }}, @{Name="lastRefreshError"; Expression={ $dataset.lastRefreshError }}, @{Name="lastRefreshErrorText"; Expression={ $dataset.lastRefreshErrorText }}, @{Name="lastRefreshErrorType"; Expression={ $dataset.lastRefreshErrorType }}, @{Name="lastRefreshErrorDetails"; Expression={ $dataset.lastRefreshErrorDetails }}, @{Name="lastRefreshErrorDetailsText"; Expression={ $dataset.lastRefreshErrorDetailsText }}, @{Name="lastRefreshErrorDetailsType"; Expression={ $dataset.lastRefreshErrorDetailsType }}, @{Name="lastRefreshErrorDetailsCount"; Expression={ $dataset.lastRefreshErrorDetailsCount }}, @{Name="lastRefreshErrorDetailsCountText"; Expression={ $dataset.lastRefreshErrorDetailsCountText }}, @{Name="lastRefreshErrorDetailsCountType"; Expression={ $dataset.lastRefreshErrorDetailsCountType }}, @{Name="lastRefreshErrorDetailsCountCount"; Expression={ $dataset.lastRefreshErrorDetailsCountCount }}, @{Name="lastRefreshErrorDetailsCountCountText"; Expression={ $dataset.lastRefreshErrorDetailsCountCountText }}, @{Name="lastRefreshErrorDetailsCountCountType"; Expression={ $dataset.lastRefreshErrorDetailsCountCountType }}, @{Name="lastRefreshErrorDetailsCountCountCount"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCount }}, @{Name="lastRefreshErrorDetailsCountCountCountText"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountText }}, @{Name="lastRefreshErrorDetailsCountCountCountType"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountType }}, @{Name="lastRefreshErrorDetailsCountCountCountCount"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCount }}, @{Name="lastRefreshErrorDetailsCountCountCountCountText"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountText }}, @{Name="lastRefreshErrorDetailsCountCountCountCountType"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountType }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCount"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCount }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountText"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountText }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountType"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountType }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCount"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCount }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountText"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountText }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountType"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountType }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountCount"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountCount }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountCountText"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountCountText }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountCountType"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountCountType }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountCountCount"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountCountCount }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountCountCountText"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountCountCountText }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountCountCountType"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountCountCountType }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCount"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCount }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountText"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountText }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountType"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountType }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCount"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCount }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountText"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountText }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountType"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountType }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountCount"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountCount }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountCountText"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountCountText }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountCountType"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountCountType }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountCountCount"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountCountCount }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountCountCountText"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountCountCountText }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountCountCountType"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountCountCountType }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountCountCountCount"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountCountCountCount }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountCountCountCountText"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountCountCountCountText }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountCountCountCountType"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountCountCountCountType }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountCountCountCountCount"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountCountCountCountCount }}, @{Name="lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountCountCountCountCountText"; Expression={ $dataset.lastRefreshErrorDetailsCountCountCountCountCountCountCountCountCountCountCountCountCountCountType }; $dsRefreshHistoryGlobal += $dsRefreshHistory
117        }
118    }
119
120    catch
121    {
122        $ex = $_.Exception
123        write-error -message "Error processing dataset: '$($ex.Message)'"
124        -erroraction continue
125    }
126
127
128
129
130
131
}
  
```

{
 "value": [
 {
 "refreshType": "ViaApi",
 "startTime": "2017-06-13T09:25:43.153Z",
 "endTime": "2017-06-13T09:31:43.153Z",
 "serviceExceptionJson": "{\"errorCode\":\"ModelRefreshFailed_CredentialsNotSpecified\"}",
 "status": "Failed",
 "requestId": "11bf290a-346b-48b7-8973-c5df149337ff"
 }
]
 }

API Throttling

- Power BI & Graph API's have throttling enabled
- Handle the exception "429 Too Many Requests"

Admin - Get Activity Events

Service: Power BI REST APIs

API Version: v1.0

Returns a list of audit activity events for a tenant.

Note: Activity logging isn't supported for Microsoft Cloud Deutschland. The user must have administrator rights (such as Office 365 Global Administrator or Power BI Service Administrator) to call this API or authenticate via service principal.

This API allows 200 requests per hour at maximum.

1 | HTTP/1.1 429 Too Many Requests
2 | Content-Type: text/html
3 | Retry-After: 3600

```
        }  
    }  
    catch [System.Net.WebException]  
{  
        $ex = $_.Exception  
        $statusCode = $ex.Response.StatusCode  
        if ($statusCode -eq 429)  
        {  
            $waitSeconds = [int]::Parse($ex.Response.Headers["Retry-After"])  
            Write-Host "429 Throttling Error - Need to wait $waitSeconds seconds..."  
            Start-Sleep -Seconds ($waitSeconds + 5)  
        }  
    }  
}
```



milestones

- ✓ IT Admin Support
- ✓ Extract Data
- ✓ Data Store
- ❑ Power BI DataSet
- ❑ Power BI Report

Power BI - PowerQuery

- Timezone offset, all dates are UTC
- Proxy Query to reference all files, easy switch data source (ex: folder or data lake)

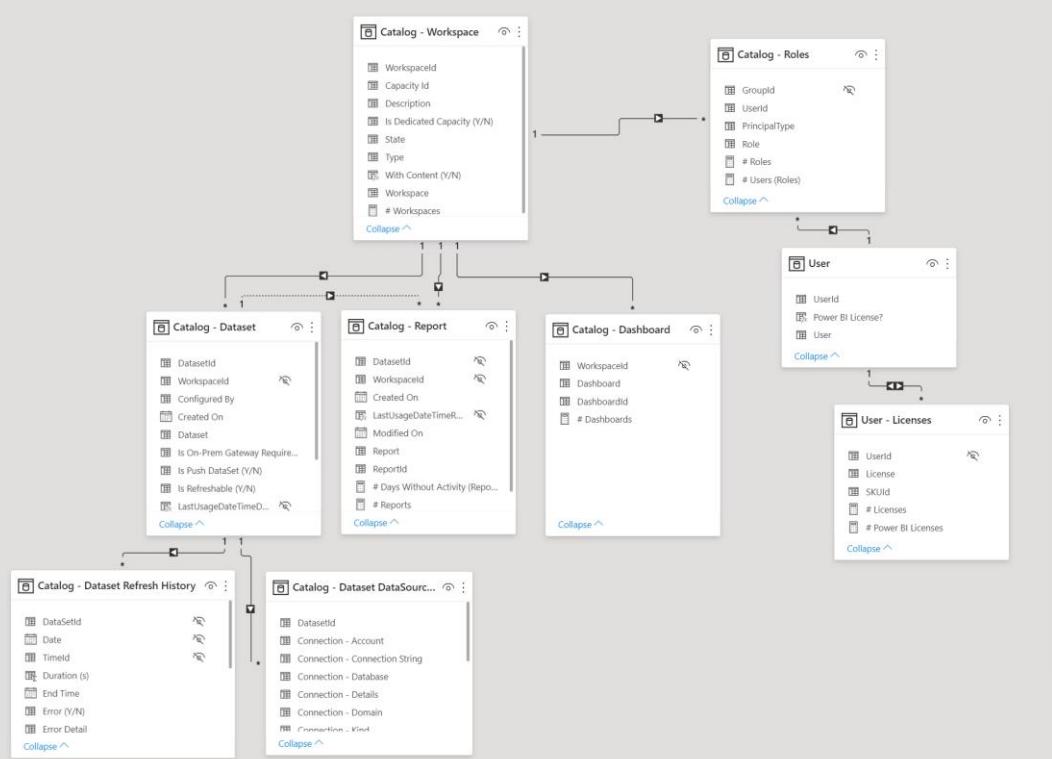
The screenshot shows the Power BI interface with the 'FilesProxy' query editor open. On the left, the 'Queries [41]' pane is visible, showing a tree structure of queries categorized under 'Catalog [13]', 'Usage [3]', 'Files - FileSystem [7]', and 'Other Queries [2]'. A red arrow points from the 'Catalog-Files' node in the tree to the 'Source' part of the query code. Another red arrow points from the 'FilesProxy' node in the tree to the 'Content' column of a preview table on the right. The preview table has columns: Content, Filename, Date, and FileType. It lists 11 rows of binary files, all of which were scanned on 10/02/2021.

Content	Filename	Date	FileType	
1	Binary	0224742b-abc1-4ddb-8e35-7cda46b31b14.json	10/02/2021	scan
2	Binary	03b400bb-0473-4e16-a5d5-62052443ca05.json	10/02/2021	scan
3	Binary	05f460dd-3944-4fb4-a607-db5c6b8a6a1d.json	10/02/2021	scan
4	Binary	09c9baf6-6df7-4bf5-b7a3-77d277527c49.json	10/02/2021	scan
5	Binary	0bef91e1-9d41-4cd3-9a43-ccc19857b41c.json	10/02/2021	scan
6	Binary	0c0fed54-fb15-4fc4-bca8-8bb062fb5259.json	10/02/2021	scan
7	Binary	0ef464ae-b606-4095-87d6-2285536300da.json	10/02/2021	scan
8	Binary	156a0679-a542-45c7-b1df-bb89181d1d86.json	10/02/2021	scan
9	Binary	18200c8c-7b91-4945-b483-e4190293ea2b.json	10/02/2021	scan
10	Binary	1a75f1d8-57f7-4d87-94db-ad656166b104.json	10/02/2021	scan
11	Binary	1b3eerb5-efda-4b37-965f-1e7fb7f50ac4.json	10/02/2021	scan

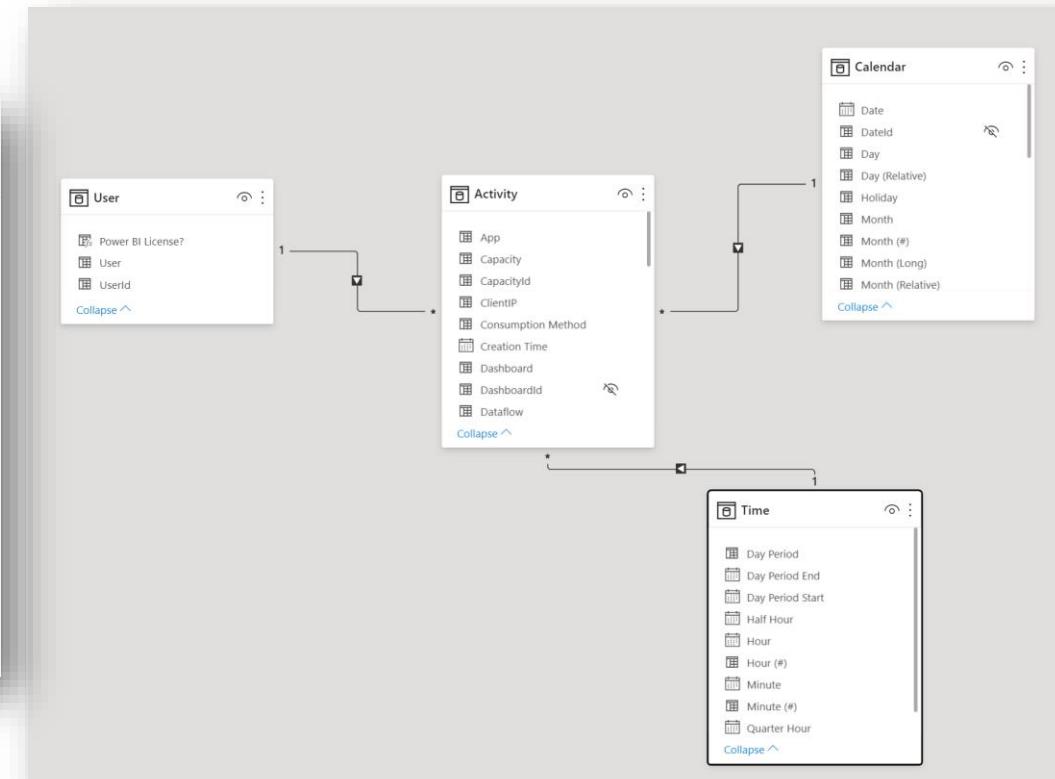
The screenshot shows the 'Queries [40]' pane on the right side of the Power BI interface. It displays a hierarchical tree of queries. The tree includes categories like 'Global Parameters [1]', 'Calendar [5]', 'Catalog [13]', 'Usage [3]', 'Users [6]', and 'Files - FileSystem [7]'. Under 'Catalog [13]', there are numerous sub-queries such as 'Catalog-Files', 'Catalog-Files-Last', 'Catalog-Scans', etc. Under 'Usage [3]', there are 'Activity-Files', 'Activity-RAW', and 'Activity'. Under 'Users [6]', there are 'Graph-Files', 'Graph-Files-Last', 'O365SKUs', 'Users - RAW', 'User', and 'User - Licenses'. Under 'Files - FileSystem [7]', there are 'DataFolder' and 'PBICatalogDataFolder'.

Power BI - DataSet, 2 Models in 1

Catalog



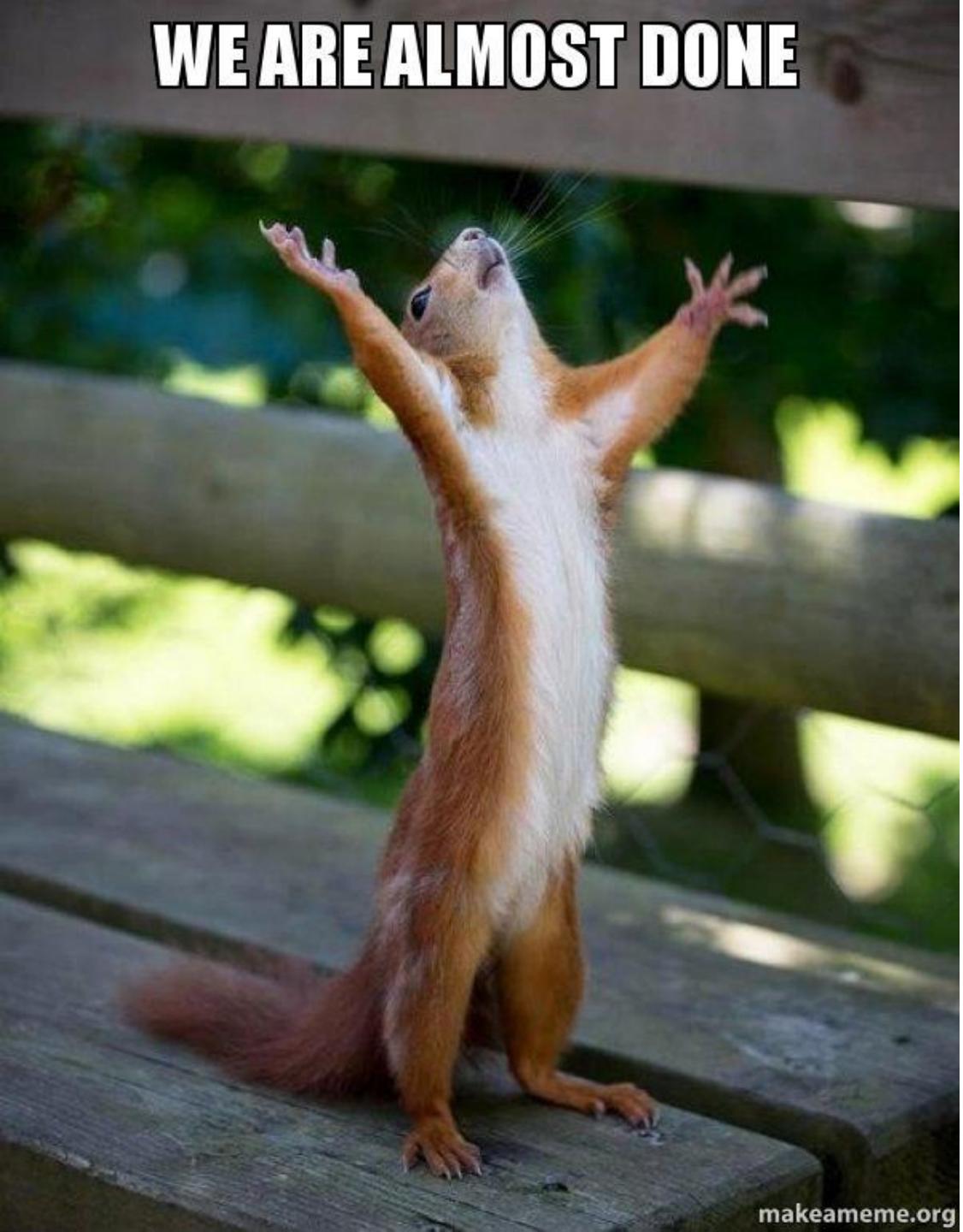
Activity



Power BI – DataSet Main Measures

Catalog	Activity
# Reports	# Logs
# DataSets	# Logs from Excel
# Reports	# Report Views
...	# Reports Created
# Days without Activity	# Distinct Users by Day
# Licenses	# Data Views
# Users	Activity Grouping => DataViews, Authoring, Admin

WE ARE ALMOST DONE



■ milestones

- ✓ IT Admin Support
- ✓ Extract Data
- ✓ Data Store
- ✓ Power BI DataSet
- ❑ Power BI Report

Power BI - Report

- Base theme, easy company branding
- Drillthrough for detail
- Look at No Activity
- Quick search for a report/dataset by id / name
- Make use of **advanced AI features:**
Explain Increase/Decrease
Anomaly Detection
Forecast



dev>scope



milestones

- ✓ IT Admin Support
- ✓ Extract Data
- ✓ Data Store
- ✓ Power BI DataSet
- ✓ Power BI Report

Rui Romano

- ❖ rui.romano@devscope.net
- ❖ linkedin.com/in/ruiromano
- ❖ @ruiromano
- ❖ <https://ruiromanoblog.wordpress.com>



dev>scope

