



# Power BI Monitoring 101

2021-06



>>>>>>>>>>>>>>>>


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://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?
- 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?

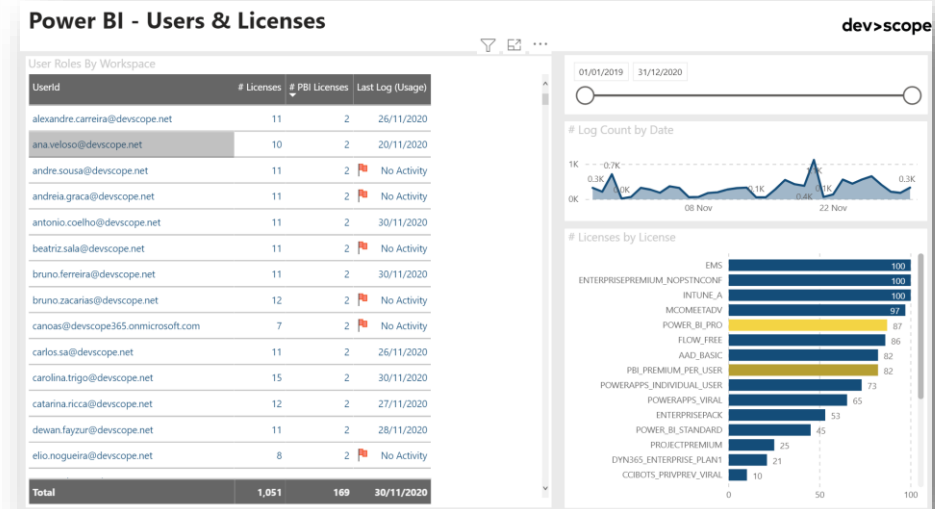
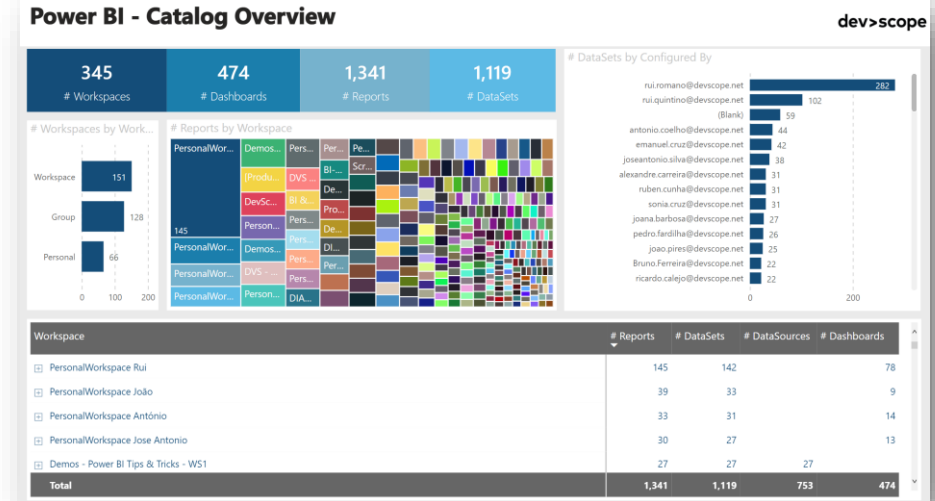
You are “driving blind” and certainly, a **BIG Reality Check** is ahead of you...

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, it's always important to do the “BI of the BI”!



# DEMO – Power BI Monitor



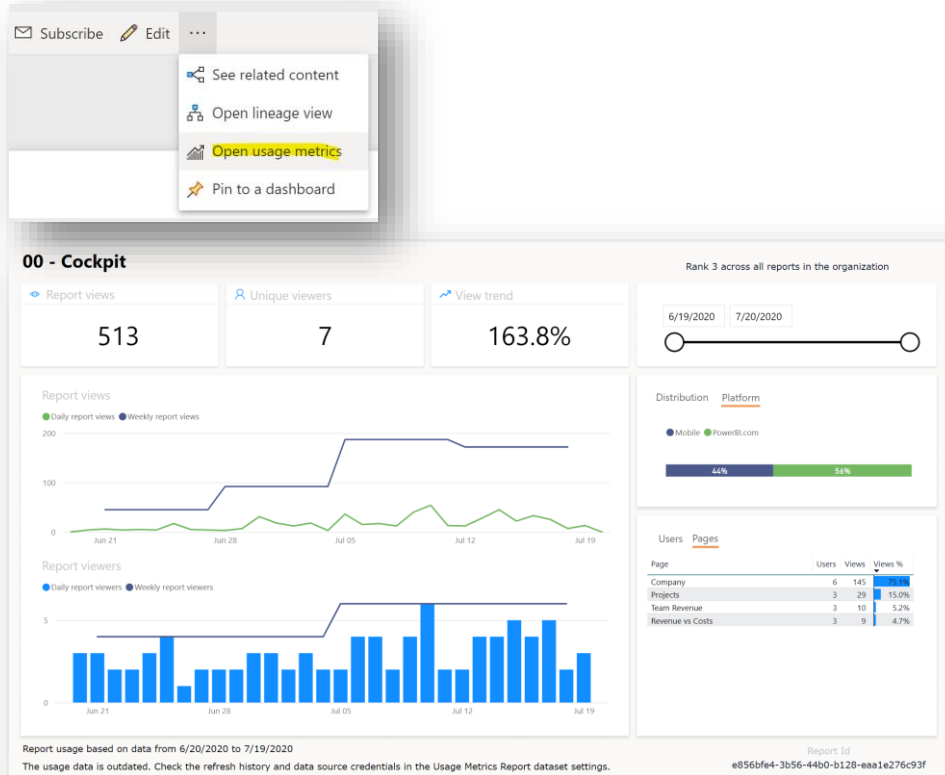
# What is available Out of the Box?

## 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

### Usage metrics

- Users
- Premium per user (preview)
- Audit logs
- Tenant settings
- Capacity settings
- Refresh summary
- Embed Codes
- Organizational visuals
- Azure connections (preview)
- Workspaces
- Custom branding
- Protection metrics
- Featured content

### Number of User Dashboards

353

Count of DashboardId

### Number of User Reports

950

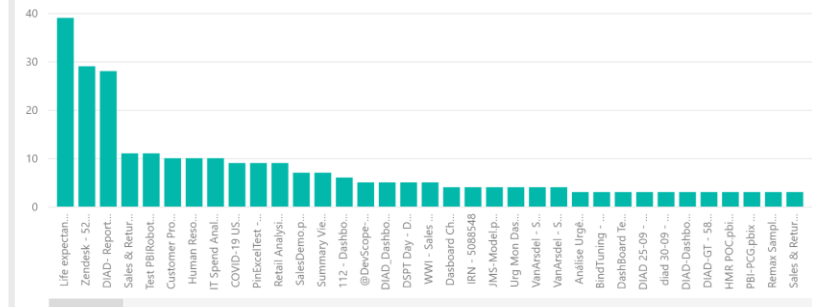
Count of Id

### Number of User Datasets

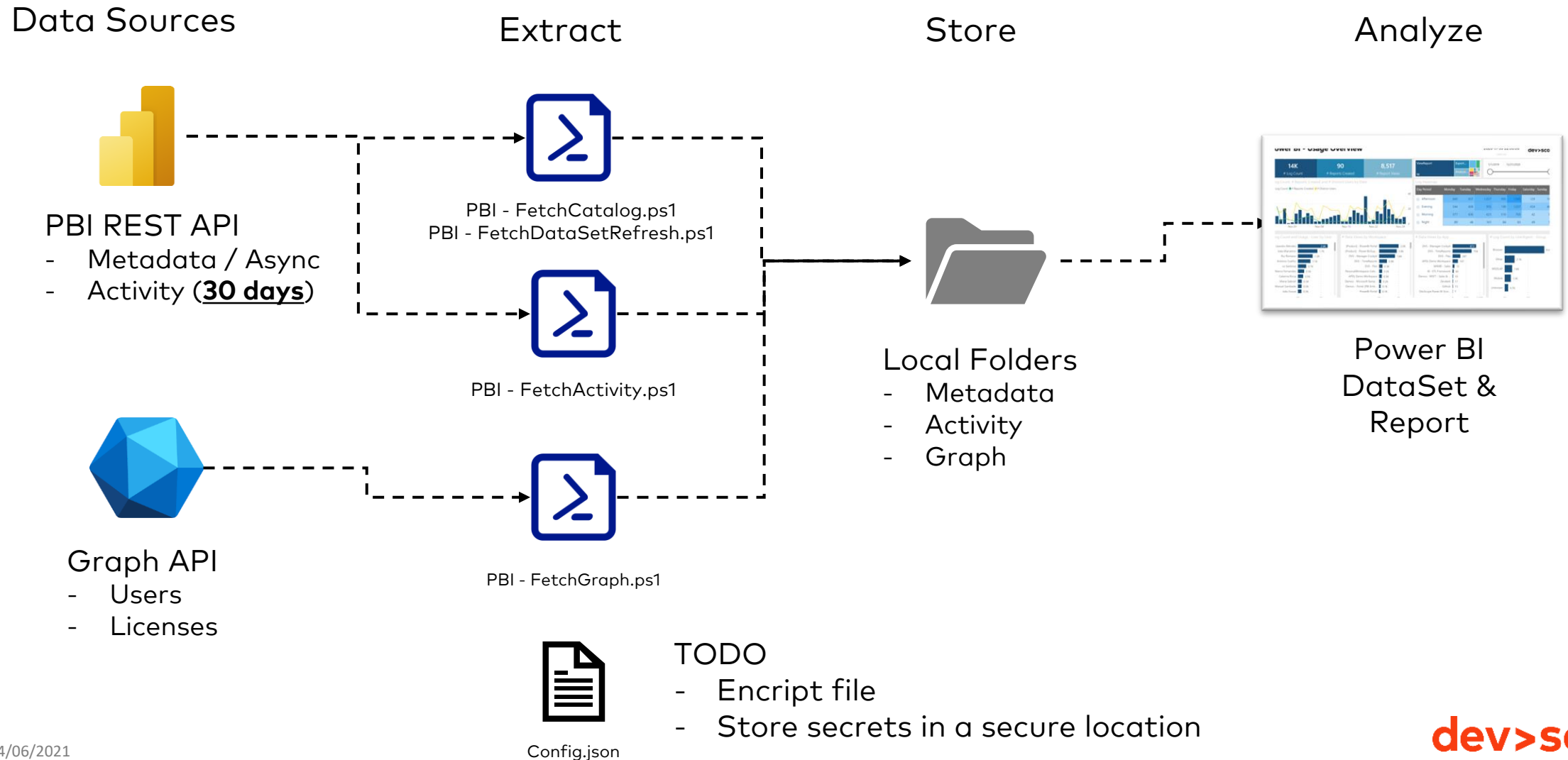
766

Count of Id

### Most Consumed Dashboards by Users



# Architecture – Simplest and Easy to Share





# Architecture – Recommended (one of many possibilities)

## Data Sources



### PBI REST API

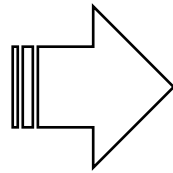
- Metadata / Async
- Audit



### Graph API

- Users
- Licenses

## Extract



### Azure Function

- Fetch Activity
- Fetch Metadata
- Fetch Users & Licenses

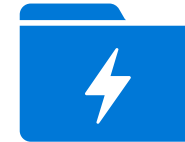
OR



### Azure Data Factory

- See Just Blindbæk [blog](#)

## Store



### Azure Storage

- \Metadata
- \Activity
- \Graph

## Analyze



### Power BI Report



Key Vault



App Insights

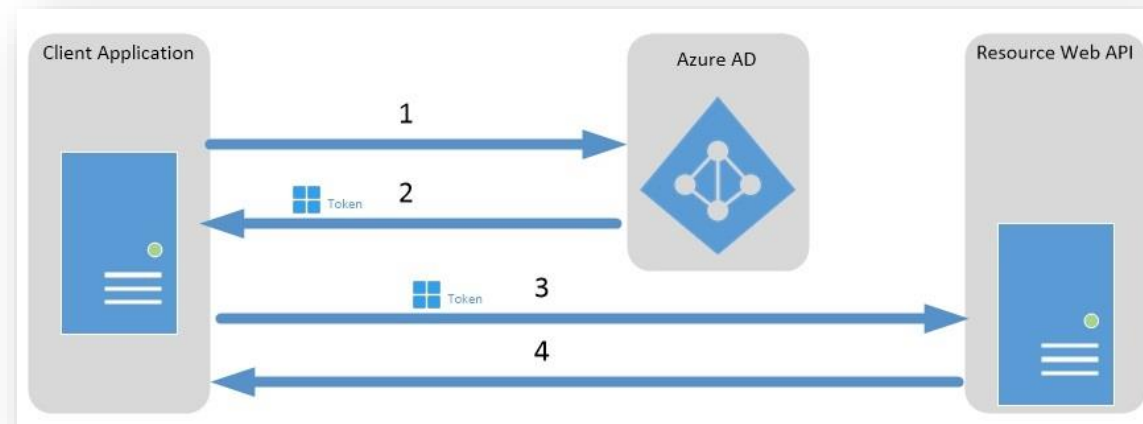
dev>scope

# API's Overview

Resource	API
Power BI Metadata <ul style="list-style-type: none"> <li>Workspaces</li> <li>DataSets</li> <li>Reports</li> <li>Dashboards</li> </ul>	<a href="#">Power BI Admin Scan APIs</a>
Power BI Metadata <ul style="list-style-type: none"> <li>Users</li> </ul>	<a href="#">Power BI Admin API – GetGroupsAsAdmin</a> + Expand Users
Power BI Metadata <ul style="list-style-type: none"> <li>RefreshHistory</li> </ul>	<a href="#">Power BI Admin API – GetGroupsAsAdmin</a> + Expand DataSets <a href="#">Power BI API - Get Refresh History In Group</a>
Power BI Activity Logs	<a href="#">Power BI Admin API - ActivityEvents</a>
Users & Licenses	<a href="#">Microsoft Graph API – Users</a>
License Info	<a href="#">Microsoft Graph API – SubscribedSKUs</a>

# API Authentication

- OAuth 2.0
- Possible Authentication Flows:



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

# Authentication Azure AD Service Principal

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

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

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

Microsoft Azure

Home > Rui Romano AD (MVP Subscription) > Demo PBI Monitor - ServicePrincipal

## Demo PBI Monitor - ServicePrincipal | Certificates & secrets

Search (Ctrl+/)

Got feedback?

Copy the new client secret value. You won't be able to retrieve it after you perform another operation or leave this blade.

Credentials enable confidential applications to identify themselves to the authentication service when receiving tokens at a web addressable location (scheme). For a higher level of assurance, we recommend using a certificate (instead of a client secret) as a credential.

**Certificates**

Certificates can be used as secrets to prove the application's identity when requesting a token. Also can be referred to as public keys.

Upload certificate

Thumbprint	Start date	Expires	ID
No certificates have been added for this application.			

**Client secrets**

A secret string that the application uses to prove its identity when requesting a token. Also can be referred to as application password.

Allow service principals to use read-only Power BI admin APIs (Preview)

*Unapplied changes*

Web apps registered in Azure Active Directory (Azure AD) will use an assigned service principal to access read-only Power BI Admin APIs without a signed in user. To allow an app to use service principal authentication, its service principal must be included in an allowed security group. By including the service principal in the allowed security group, you're giving the service principal read-only access to all the information available through Power BI admin APIs, including names and emails of Power BI users. [Learn more](#)

☒ Enabled

Apply to:

☐ The entire organization

☒ Specific security groups

Demo PBI Monitor - Service Principals X Enter security groups

Apply Cancel



## 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

```
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     $pivotDate = [datetime]::UtcNow.Date.AddDays(-30)
39 }
40
41 # Gets audit data daily
42 while($pivotDate -le [datetime]::UtcNow)
43 {
44     Write-Host "Getting audit data for: '$($pivotDate.ToString("yyyyMMdd"))'"
45     $odataParams = "startDateTime='$($pivotDate.ToString("s"))'&endDateTime='$($pivotDate.AddHours(23).ToString("s"))'"
46     $audits = @(Invoke-PBIRestRequest -authToken $authToken -resource "activityevents" -admin -odata $odataParams)
47     if ($audits.Count -gt 0)
48     {
49         Write-Host "'$($audits.Count)' audits"
50         $outputFilePath = "$outputPath\$($pivotDate.ToString("yyyyMMdd")).json"
51         ConvertTo-Json $audits | Out-File $outputFilePath -force
52     }
53     else
54     {
55         Write-Warning "No audit logs for date: '$($pivotDate.ToString("yyyyMMdd"))'"
56     }
57     $config.Activity.LastRun = [datetime]::UtcNow.Date.ToString("o")
58     $pivotDate = $pivotDate.AddDays(1)
59     # Save config
60     ConvertTo-Json $config | Out-File $configPath
61 }
62 }
```

```
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>
```

```
1 {
2   "ServicePrincipal": {
3     "AppId": "7f9f1c82-8d4d-4a8b-98ea-5891447f0000"
4     "AppSecret": "AaBbCcDdEeFfGgHhIiJjKkLlMmNnOoPpQqRrSsTtUuVvWwXxYyZz"
5     "TenantId": "72f988bf-86f1-41af-91ab-2d7cd011db47"
6   },
7   "Catalog": {
8     "LastRun": "2021-02-18T00:00:00.0000000Z"
9   },
10  "Activity": {
11    "LastRun": "2021-02-18T00:00:00.0000000Z"
12  }
13 }
```

ub > pbimonitor > Data > Usage

Name
20210114.json
20210116.json
20210117.json
20210130.json
20210201.json
20210207.json
20210209.json
20210211.json

# 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-PBIRestRequest -authToken $authToken -resource "workspaces" -body $bodyStr
            $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 $config.ServicePrincipal.Secret
        }
    }
}
while($workspacesBatch.Count -ne 0 -and $workspacesScanRequests.Count -lt 5000)
```

```
PS C:\@Repos\Github\pbimonitor> C:\@Repos\Github\pbimonitor\Scripts\FetchCatalog.ps1
Fetching 5000 /admin/workspaces
Getting workspaces to scan
Since: 2021-02-18T00:00:00.000000Z
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 - 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     $item++
99
100     Write-Host "Processing workspace: '$($workspace.Name)' $item/$total"
101
102     Write-Host "Datasets: $($workspace.datasets.Count)"
103
104     $refreshableDatasets = @($workspace.datasets |? { $_.isRefreshable -eq $true -and $_.addRowsAPIEnabled -eq $false})
105
106     Write-Host "Refreshable Datasets: $($refreshableDatasets.Count)"
107
108     foreach($dataset in $refreshableDatasets)
109     {
110         try
111         {
112             Write-Host "Processing dataset: '$($dataset.name)'"
113
114             Write-Host "Getting refresh history"
115
116             $dsRefreshHistory = Invoke-PBIRest -authToken $authToken -resource "datasets/$($dataset.id)/refreshes" -groupId $workspace.id
117
118             if ($dsRefreshHistory)
119             {
120                 $dsRefreshHistory = $dsRefreshHistory | Select *, @{Name="datasetId"; Expression={ $dataset.id }}, @{Name="dataset"; Expression={ $dataset.name }}, @{Name="group"; Expression={ $workspace.name }}, @{Name="configuredBy"; Expression={ $dataset.configuredBy }}
121
122                 $dsRefreshHistoryGlobal += $dsRefreshHistory
123             }
124         }
125         catch
126         {
127             $ex = $_.Exception
128
129             Write-Error -message "Error processing dataset: '$($ex.Message)'" -ErrorAction Continue
130
131         }
132     }
133 }

```

```

{
  "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"
    }
  ]
}

```

# Script – PBI - FetchGraph.ps1

- Snapshot tenant:
  - Users & Assigned Licenses
  - Tenant Subscribed SKUs
- Uses the Microsoft Graph API
  - [Users](#)
  - [SubscribedSkus](#)
- Paginated API

```
{
  "@odata.context": "https://graph.microsoft.com/v1.0/$metadata#users",
  "@odata.nextLink": "https://graph.microsoft.com/v1.0/users?
$skiptoken=X%2744537074020001000000163A62656C6F2E62656C6C756D40676D61696C2E636F6D295
  "value": [
    {
```

```
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,capability"
$filePath = "$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

# 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 Throthling Error - Need to wait $waitSeconds seconds..."
        Start-Sleep -Seconds ($waitSeconds + 5)
        $authToken = Get-PBIAuthToken -clientId $config.ServicePrincipal.AppId -clientSecret $config.ServicePrincipal.ClientSecret
    }
}
```



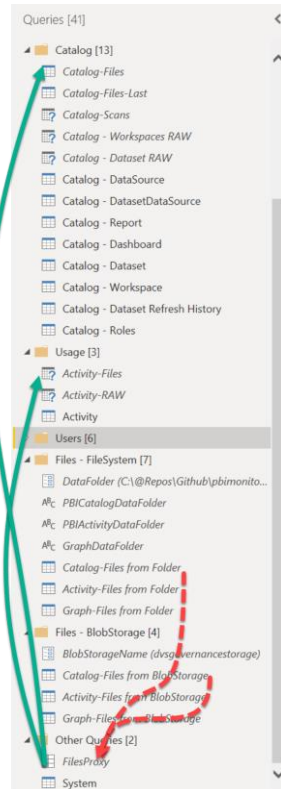


## milestones

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

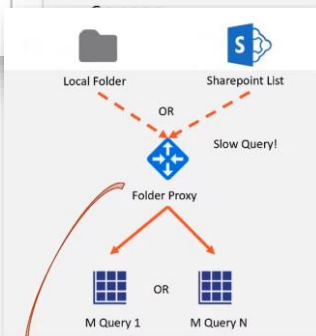
# Power BI - PowerQuery

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

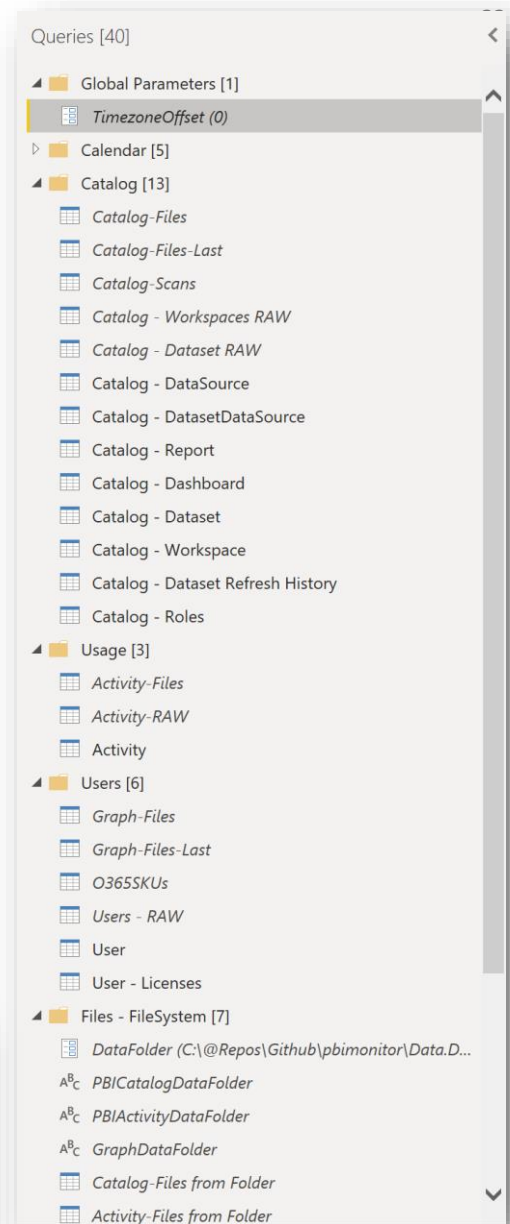


## FilesProxy

```
let
    Source = [ActivityFiles = #"Activity-Files from Folder"
              , CatalogFiles = #"Catalog-Files from Folder"
              , GraphFiles = #"Graph-Files from Folder"
            ]
    //Source = [ActivityFiles = #"Activity-Files from BlobStorage"
    //, CatalogFiles = #"Catalog-Files from BlobStorage"
    //, GraphFiles = #"Graph-Files from BlobStorage"
    //]
in
```

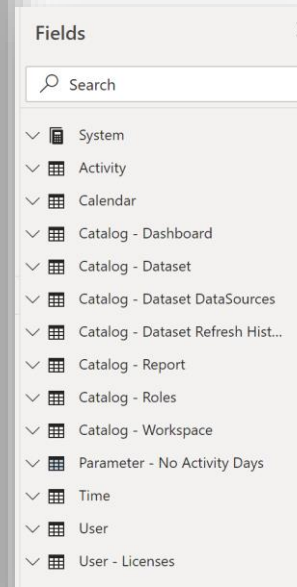
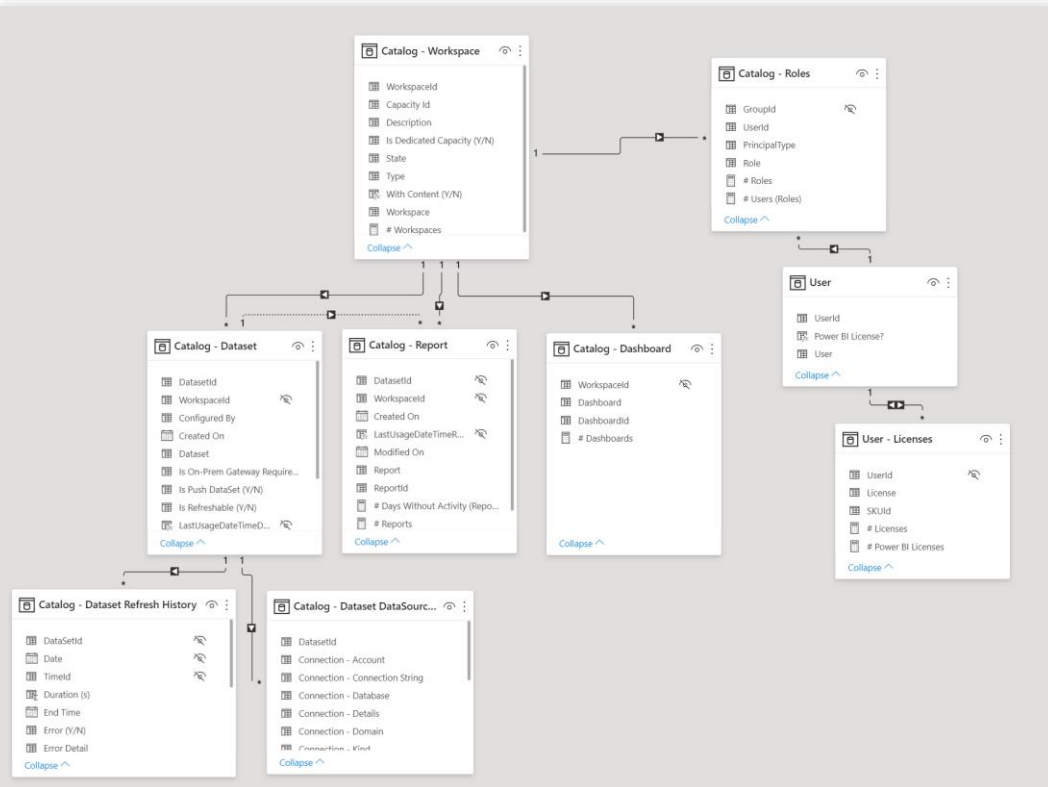


	Content	Filename	Date	FileType
1	Binary	0224742b-abc1-4ddb-8e35-7cda46b31b14.json	10/02/2021	scan
2	Binary	03b400bb-0473-4e16-a5d5-62052443ca05.json	10/02/2021	scan
3	Binary	05f460dd-3944-4fbd-a607-db5c6b8a6a1d.json	10/02/2021	scan
4	Binary	09c9baf6-6df7-4bf5-b7a3-77d277527c49.json	10/02/2021	scan
5	Binary	0bef91e1-9d41-4cd3-9a43-ccc19857b41c.json	10/02/2021	scan
6	Binary	0c0fed54-fb15-4fcd-bca8-8bb062fb5259.json	10/02/2021	scan
7	Binary	0ef464ae-b606-4095-87d6-228553630dda.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	1b3eech5-efda-4b37-965f-1e7fb7f50ac4.json	10/02/2021	scan

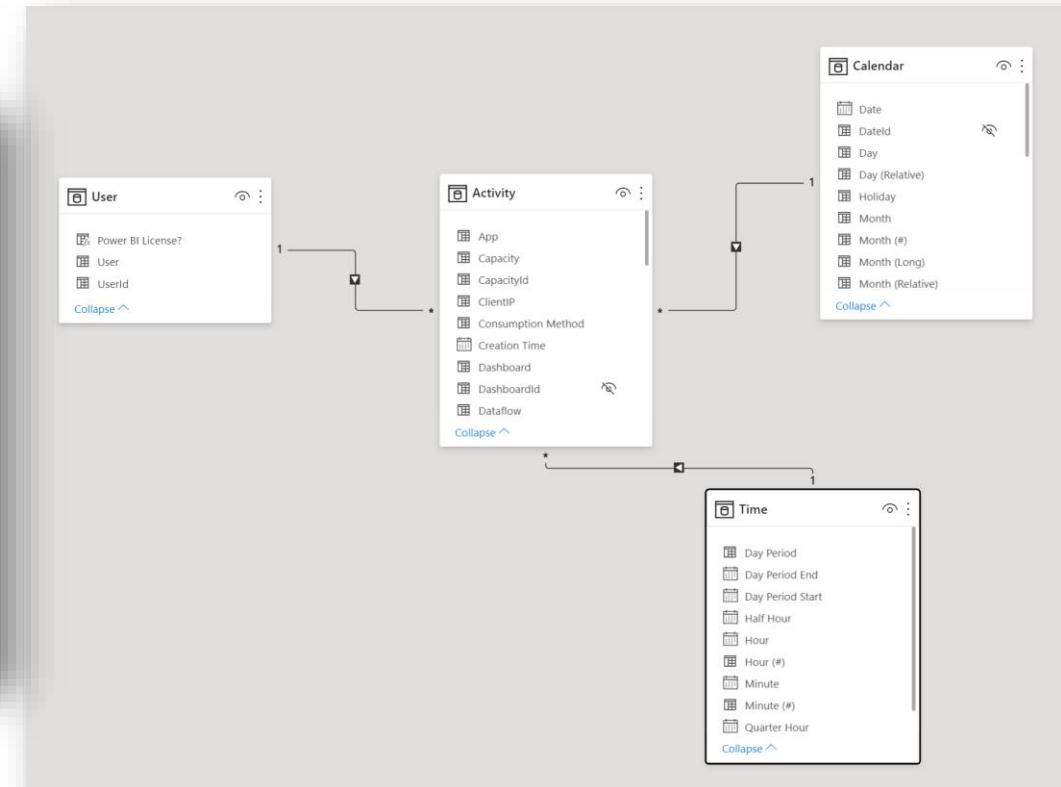


# Power BI – DataSet, 2 Models in 1

## Catalog



## Activity



# Power BI – DataSet Main Measures

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

**WE ARE ALMOST DONE**



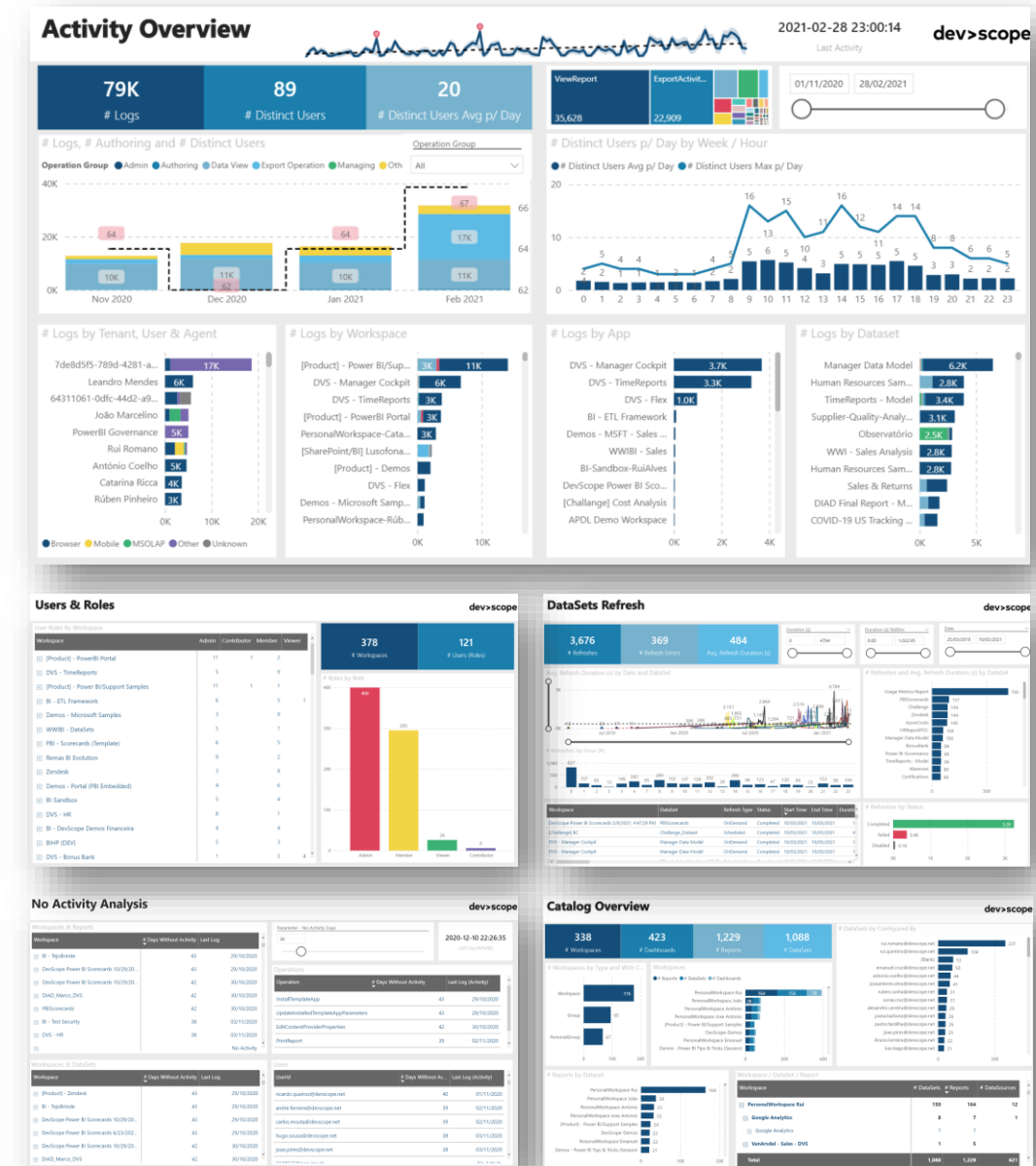
## milestones

- ✓ 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





## milestones

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

# GIVEAWAY!

5x



PowerBI Tiles Pro

REDEEM CODE

**PBITiles\_M365Chicago**

GO TO:

**[powerbitiles.com/redeem](https://powerbitiles.com/redeem)**



## Rui Romano

- ❖ [rui.romano@devscope.net](mailto:rui.romano@devscope.net)
- ❖ [linkedin.com/in/ruiromano](https://www.linkedin.com/in/ruiromano)
- ❖ [@ruiromano](https://twitter.com/ruiromano)
- ❖ <https://ruiromanoblog.wordpress.com>

# Thanks!

dev>scope

