

# No more data quality surprises!

Azure Databricks  
Synapse Analytics  
Power BI



# Dave Ruijter

Solution Architect Data & Analytics  
Blue Rocket IT



[dave@blue-rocket.it](mailto:dave@blue-rocket.it)



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



[linkedin.com/in/DaveRuijter](https://linkedin.com/in/DaveRuijter)



[ModernData.ai](https://ModernData.ai)

**THERE ARE NO DATA ISSUES**

**IF YOU DON'T CHECK DATA QUALITY**





THIS IS  
*FINE*

# What is data quality

- Inputs are wrong
- Your understanding is wrong
- Your understanding is out-dated



great  
expectations

# What can Great Expectations do for you

- Data validation
- Documentation
- Data profiling

# Approach

- Define your expectations
- Add validation step(s)
- Feedback loop





# What are expectations?

- `expect_column_to_exist`
- `expect_table_row_count_to_be_between`
- `expect_column_values_to_be_unique`
- `expect_column_values_to_not_be_null`
- `expect_column_values_to_be_between`
- `expect_column_values_to_match_regex`
- `expect_column_kl_divergence_to_be_less_than`

# Glossary of expectations

## Dataset

Dataset objects model tabular data and include expectations with row and column semantics. Many Dataset expectations are implemented using `column_map_expectation` and `column_aggregate_expectation` decorators.

Not all expectations are currently available for each backend. A table describing available implementations per-backend is available here: [Table of Expectation Implementations By Backend](#).

## Table shape

- `expect_column_to_exist`
- `expect_table_columns_to_match_ordered_list`
- `expect_table_row_count_to_be_between`
- `expect_table_row_count_to_equal`

## Missing values, unique values, and types

- `expect_column_values_to_be_unique`
- `expect_column_values_to_not_be_null`
- `expect_column_values_to_be_null`
- `expect_column_values_to_be_of_type`
- `expect_column_values_to_be_in_type_list`

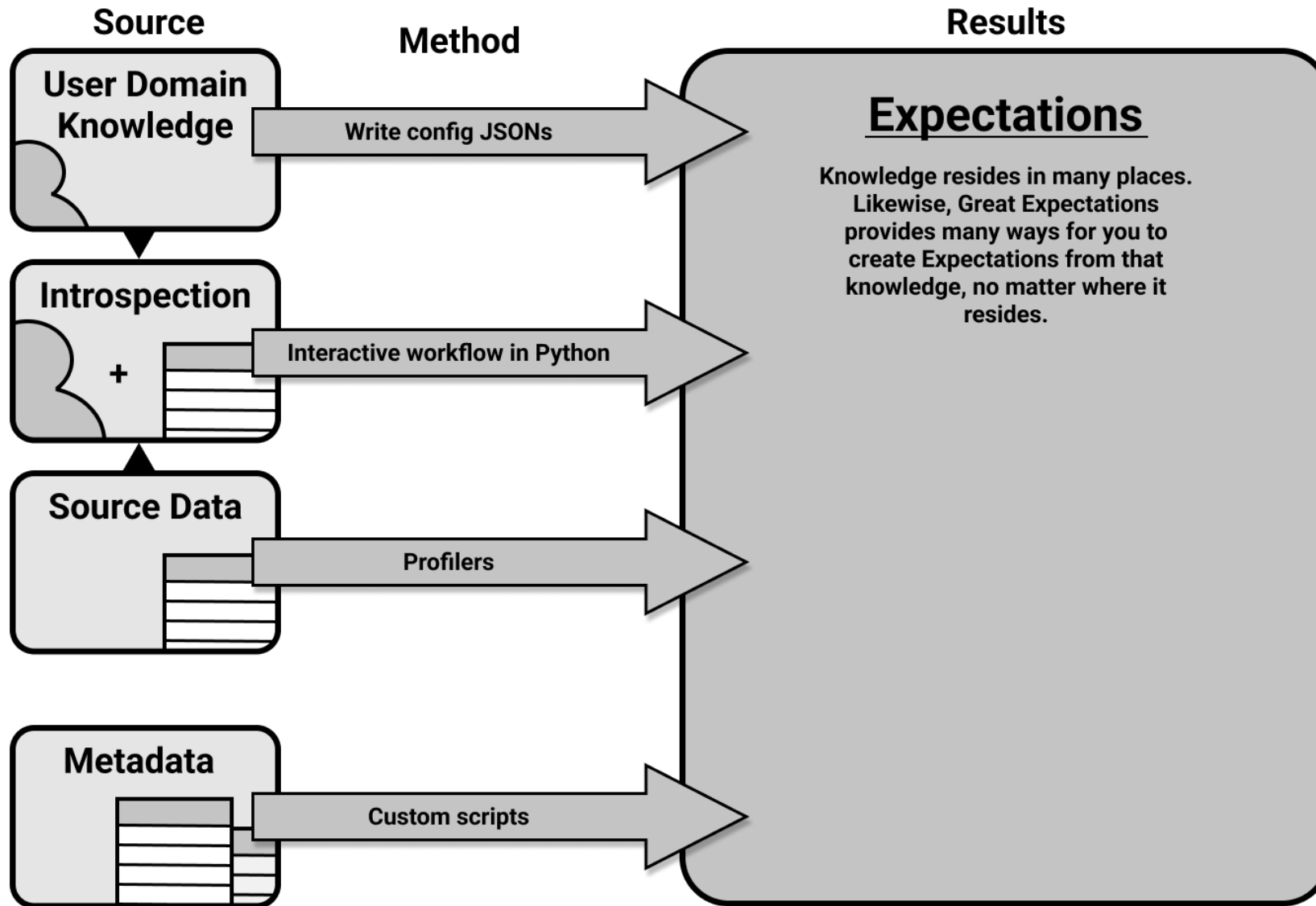
# JSON example of expectation #1

```
1  {  
2    "expectation_type": "expect_column_values_to_not_be_null",  
3    "kwargs": {  
4      "column": "id"  
5    }  
6  }
```

# JSON example of expectation #2

```
1  {
2    "expectation_type": "expect_column_values_to_match_regex",
3    "kwargs": {
4      "column": "firstName",
5      "regex": "[A-Za-z0-9\\.\\,\\;\\:?!?()\\\"'%'\\-]+"
6    }
7  }
```

# Where do Expectations come from?



# Libraries in cluster

<input type="checkbox"/>	Name	Type	Status
<input type="checkbox"/>	great-expectations	PyPI	✔ Installed
<input type="checkbox"/>	azure.identity	PyPI	✔ Installed
<input type="checkbox"/>	azure-storage-blob	PyPI	✔ Installed



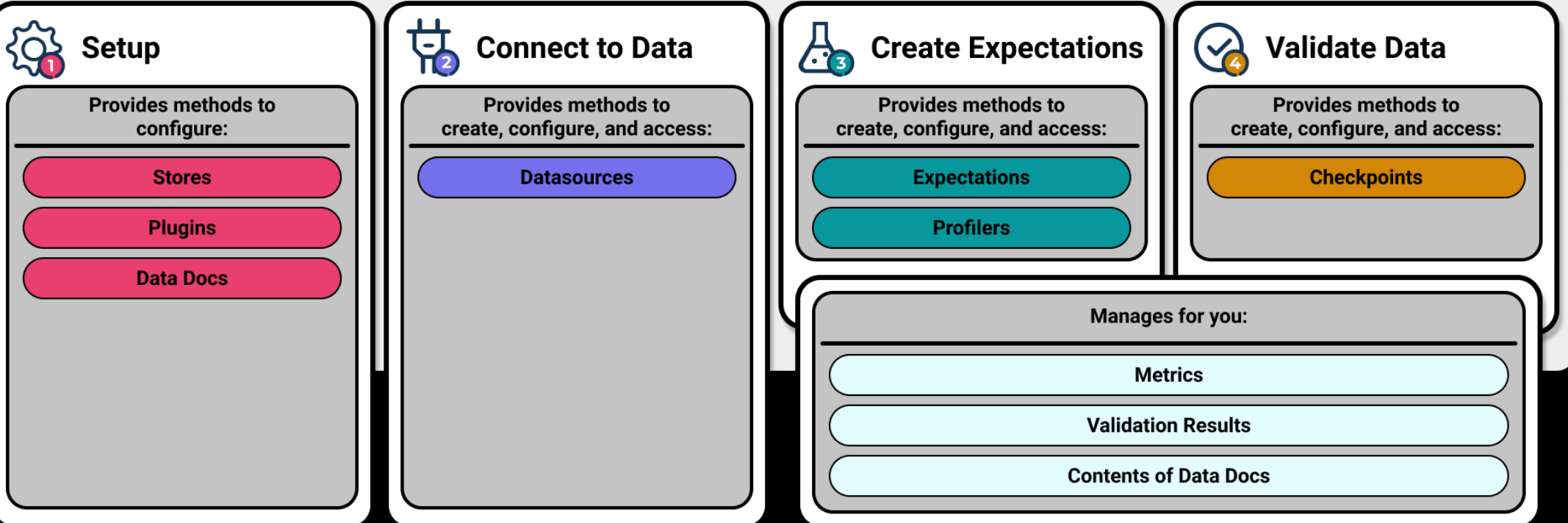
# Demo

Intro into great-expectations (GX)



# The Data Context

## What your **Data Context** does for you throughout using Great Expectations



# Demo

- Using Data Context
- Creating expectation suits
- Run validations



# Demo

Workflow integration



# Taking action

- Investigate pipeline runs
- Validate GX results ([DataDocs](#))
- Fix data
- Change expectation

# Validation results save you time.

Status	Expectation	Observed Value												
✓	values must never be null.	100% not null												
✗	values must belong to this set: <b>Y</b> <b>N</b> .	≈20.08% unexpected												
<b>218612 unexpected values found. ≈20.08% of 1088696 total rows.</b>														
<table><tr><th>Unexpected Value</th><th>Count</th></tr><tr><td>yes</td><td>10</td></tr><tr><td>no</td><td>4</td></tr><tr><td>No.</td><td>2</td></tr><tr><td>n</td><td>2</td></tr><tr><td>y</td><td>2</td></tr></table>			Unexpected Value	Count	yes	10	no	4	No.	2	n	2	y	2
Unexpected Value	Count													
yes	10													
no	4													
No.	2													
n	2													
y	2													



# DataDocs on Azure

<https://dlswedataplatformlab.z6.web.core.windows.net/index.html>

# Taking action

result\_format:

- BOOLEAN\_ONLY
- BASIC
- SUMMARY
- COMPLETE

Fields within <code>result</code>	BOOLEAN_ONLY	BASIC	SUMMARY	COMPLETE
observed_value	no	yes	yes	yes
details (e.g. statistical details)	no	no	yes	yes

[https://docs.greatexpectations.io/docs/reference/expectations/result\\_format/](https://docs.greatexpectations.io/docs/reference/expectations/result_format/)

# Row identifiers

```
1 result_format_dict: dict = {  
2     "result_format": {  
3         "result_format": "COMPLETE",  
4         "unexpected_index_column_names": ["event_id"],  
5     }  
6 }
```

```
result = {  
    unexpected_index_list = [  
        {'event_type': 'user_signup', 'event_id': 3},  
        {'event_type': 'purchase', 'event_id': 4},  
        {'event_type': 'download', 'event_id': 5}  
    ]  
}
```

# Validation produces a validation result object

```
{
  "expectation_config": {
    "expectation_type": "expect_column_values_to_not_be_null",
    "kwargs": {
      "column": "user_id"
    },
  },
  "success": false,
  "result": {
    "element_count": 253405,
    "unexpected_count": 7602,
    "unexpected_percent": 2.999
  },
}
```

# Power BI data validations



# Example Power BI data validations

- *Verify the exact value of an order*
- *Verify sum of a measure for 2020, 2021*
- *Verify complex measures*
  - *Percentages*
  - *Time intelligence*
  - *Period over period*
- *Verify if all dimension values (e.g. product brands) are accounted for*





Filter by title

- Overview
- > Admin
- > Apps
- > Available Features
- > Capacities
- > Dashboards
- > Dataflow Storage Accounts
- > Dataflows
- ▼ Datasets
  - Overview
  - Bind To Gateway
  - Bind To Gateway In Group
  - Cancel Refresh
  - Cancel Refresh In Group
  - Delete Dataset
  - Delete Dataset In Group
  - Discover Gateways
  - Discover Gateways In Group
  - Execute Queries**
  - Execute Queries In Group
  - Get Dataset
  - Get Dataset In Group
  - Get Dataset To Dataflows Links In Group
  - Get Dataset Users
  - Get Dataset Users In Group
  - Get Datasets
  - Get Datasets In Group
  - Get Datasources

[Learn](#) / [Power BI REST APIs](#) / [Datasets](#) /



# Datasets - Execute Queries

Reference

[Feedback](#)

Service: [Power BI REST APIs](#)

API Version: [v1.0](#)

Executes Data Analysis Expressions (DAX) queries against the provided dataset. The dataset must reside in **My workspace** or another workspace.

DAX query errors will result in:

A response error, such as `DAX query failure`.

A failure HTTP status code (400).

A query that requests more than one table, or more than the allowed number of table rows, will result in:

Limited data being returned.

A response error, such as `More than one result table in a query` or `More than {allowed number} rows in a query result`.

A successful HTTP status code (200).

Columns that are fully qualified in the query will be returned with a fully qualified name, for example, `MyTable[MyColumn]`. Columns that are renamed or created in the query will be returned within square bracket, for example, `[MyNewColumn]`.

## Permissions

The user must have [Manage dataset access permissions](#).

## Required Scope

`Dataset.ReadWrite.All` or `Dataset.Read.All`

# Execute queries API

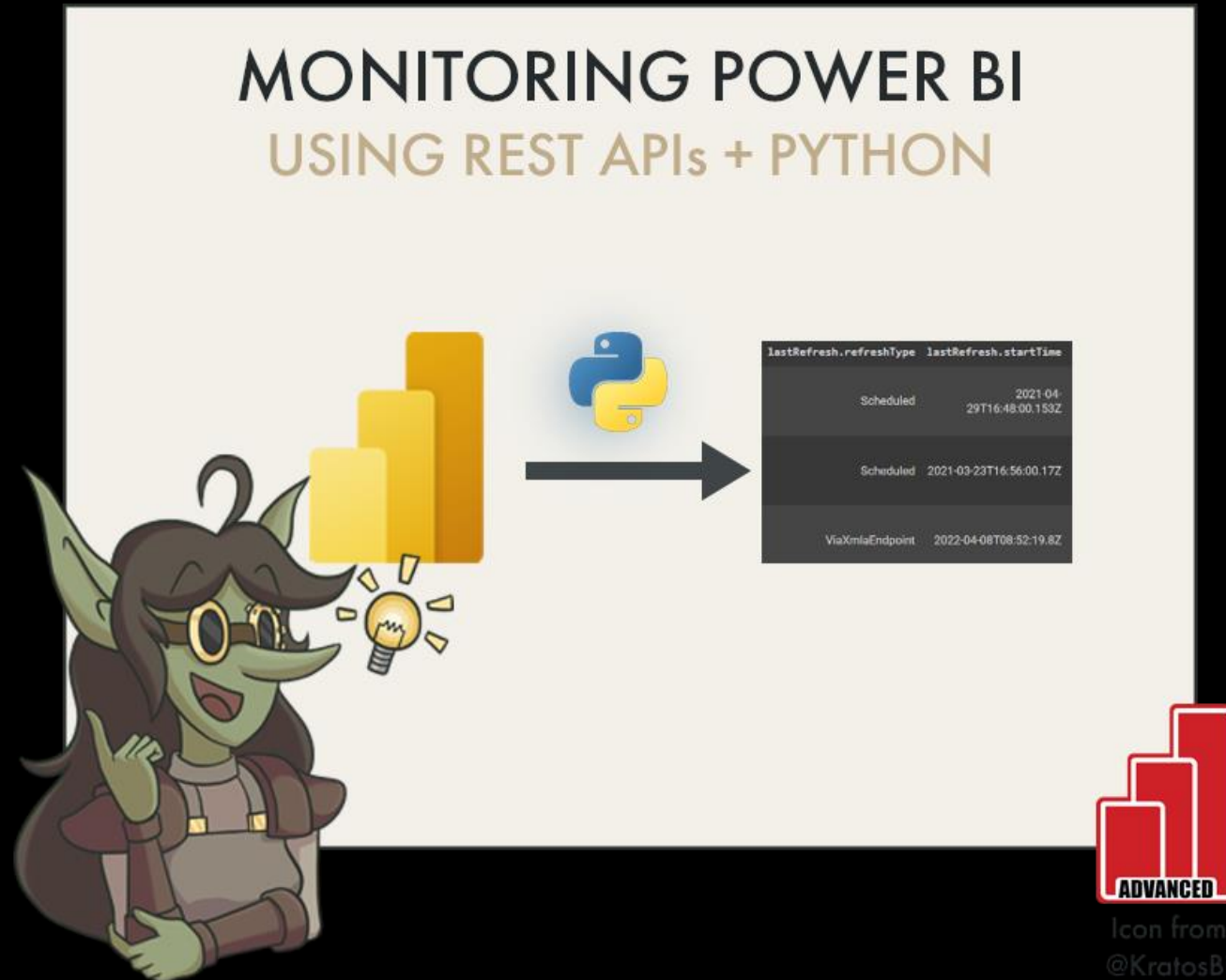
- Testing the data model, not the reports
- On datasets in the Power BI Service (not locally in Desktop)
- Does *not* require Premium license/capacity

<https://docs.microsoft.com/en-us/rest/api/power-bi/datasets/execute-queries>





<https://data-goblins.com/power-bi/power-bi-api-python>



DATA GOBLINS



Icon from  
@KratosBi

# Example API call message

```
{  
  "queries": [  
    {  
      "query": "EVALUATE SUMMARIZECOLUMNS(Products[Brand Class])"  
    }  
  ],  
  "serializerSettings": {  
    "includeNulls": true  
  },  
  "impersonatedUserName": "someuser@mycompany.com"  
}
```



# Example API call result

```
{  
  "results": [{  
    "tables": [{  
      "rows": [  
        {"Products[Brand Class]": "NON-MAGIC PRODUCTS"},  
        {"Products[Brand Class]": "MAGIC PRODUCTS"},  
        {"Products[Brand Class]": "MAGIC BRANDS"},  
        {"Products[Brand Class]": "NON-MAGIC BRANDS"}  
      ]  
    }  
  ]  
}]}
```

# What do we need?

1. Service Principal (aka App Registration)
2. Tenant setting enabled
3. Workspace/dataset permissions

# Setup

## Integration settings

- ▶ Allow XMLA endpoints and Analyze in Excel with on-premises datasets

*Enabled for the entire organization*

- ▲ Dataset Execute Queries REST API

*Enabled for the entire organization*

Users in the organization can query datasets by using Data Analysis Expressions (DAX) through Power BI REST APIs.

☒ Enabled

Apply to:

- ☒ The entire organization
- ☐ Specific security groups
- ☐ Except specific security groups

Apply

Cancel

## Admin API settings

- ▲ Allow service principals to use read-only admin APIs

*Enabled for a subset of the organization*

Web apps registered in Azure Active Directory (Azure AD) will use an assigned service principal to access read-only 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 admin APIs (current and future). For example, user names and emails, dataset and report detailed metadata. [Learn more](#)

☒ Enabled

Apply to:

- ☐ The entire organization
- ☒ Specific security groups

PBI\_Service\_API\_Permissions X Enter security groups

Apply



Cancel

# Setup

 Refresh |  Got feedback?

## Configured permissions

Applications are authorized to call APIs when they are granted permissions by users/admins as part of the consent process. The list of configured permissions should include all the permissions the application needs. [Learn more about permissions and consent](#)

 Add a permission  Grant admin consent for Dave Ruijter

API / Permissions name	Type	Description	Admin consent requ...	Status
▼ Microsoft Graph (1) ...				
User.Read	Delegated	Sign in and read user profile	No	✔ Granted for Dave Ruijter ...
▼ Power BI Service (1) ...				
Dataset.ReadWrite.All	Delegated	Read and write all datasets	No	✔ Granted for Dave Ruijter ...

# Demo

Validate Power BI data



# Limitations

- Maximum of 120 requests per user per minute.
- One query per API call. One table request per query.
- Service Principals aren't supported for datasets with RLS
- Maximum of 100,000 rows or 1,000,000 values per query (whichever is hit first). For example if you query for 5 columns, you can get back max 100,000 rows. If you query for 20 columns, you can get back max 50,000 rows (1 million divided by 20).
- Maximum of 15MB of data per query. Once 15MB is exceeded, the current row will be completed but no additional rows will be written.

- ❖ [Monitoring Power BI using REST APIs from Python](#)  
[DATA GOBLINS \(data-goblins.com\)](#)
- ❖ [Announcing general availability of the ExecuteQueries REST API |](#)  
[Microsoft Power BI Blog | Microsoft Power BI](#)

# Take-aways on great\_expectations

- Impressive
- Timesaver
- Configuration nightmare





# Best practices

- Test early, test often
- Start small, start basic
- Focus on critical elements, focus on high impact
- Involve experts
- Create standard operating procedures
- Use data contracts















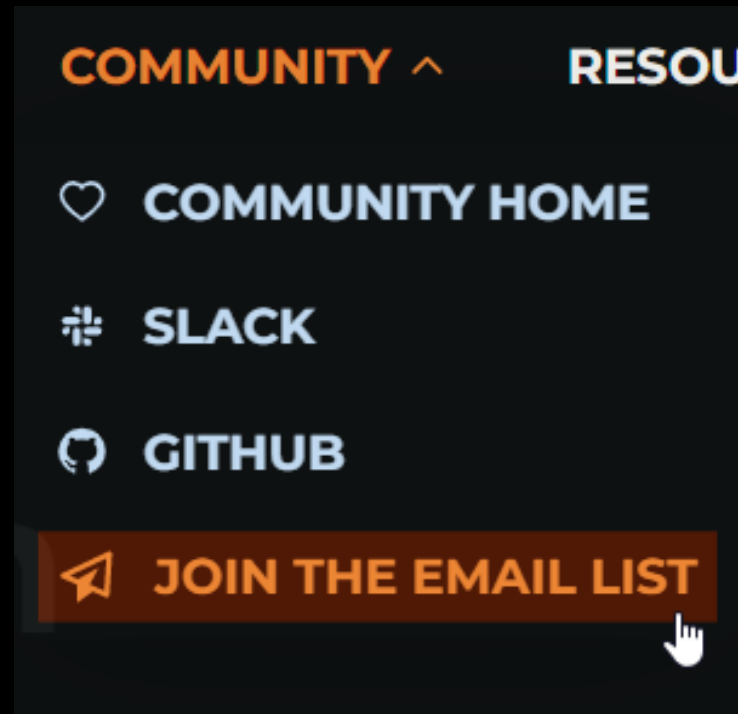






# Next steps

- Join the GX Slack channel: <https://greatexpectations.io/slack>
- Join the GX email list:







Please provide session feedback:



<https://sqlb.it/?9862>