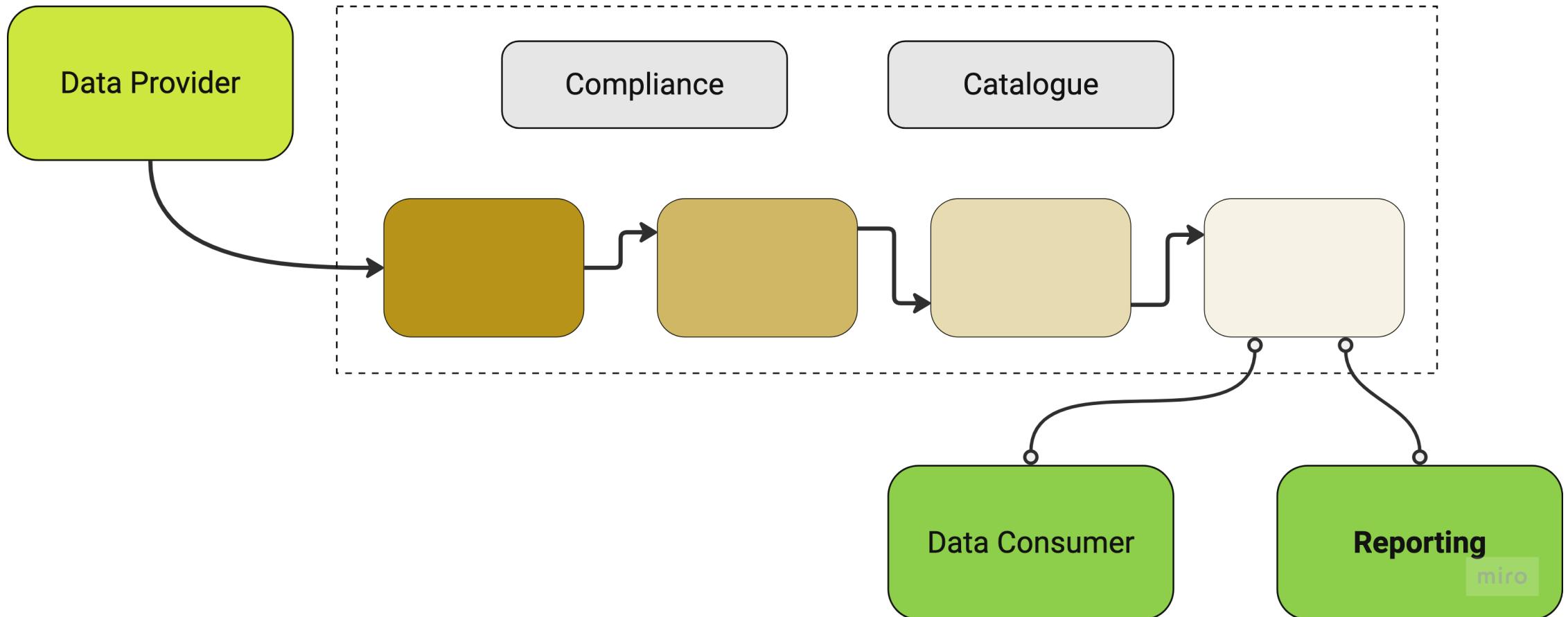


# Applying engineering principles to reporting

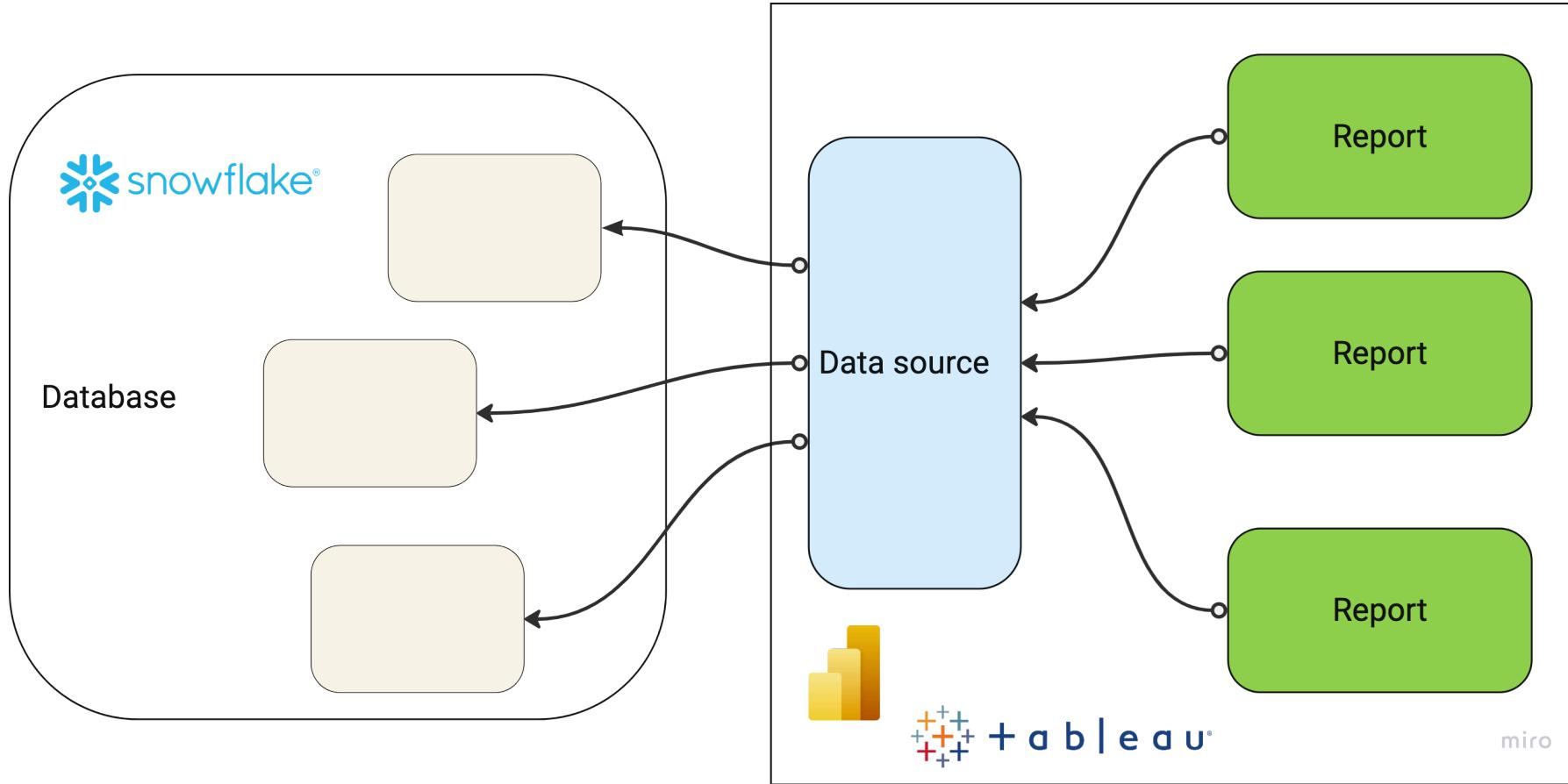
Guillaume (**G**) Belrose



# Data Pipelines: What | How



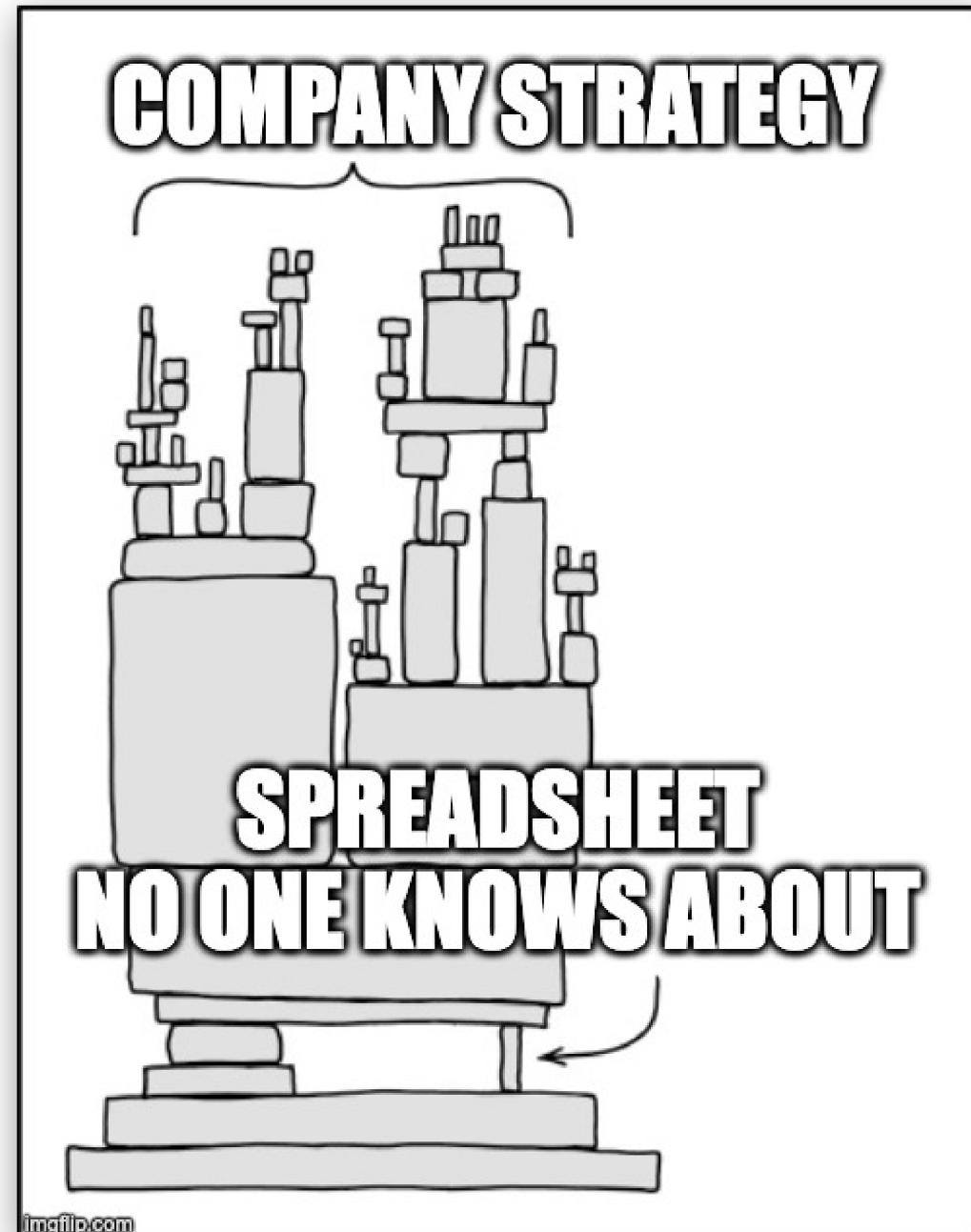
# Reporting 101



# Agenda

**Discovery, extend right & shift left**

## Discovery



# Tableau lineage

The screenshot shows the Tableau lineage interface for a workbook named "Tableau (Tableau practice)".

**Left Sidebar:**

- Home
- Favourites
- Recents
- Shared with Me
- Recommendations
- Personal Space
- Collections
- Explore** (selected)
- External Assets

**Top Bar:**

- Tableau (Tableau practice) (Icon)
- Owner [REDACTED]
- Data is Extract ▾ 21 May 2024, 14:46
- New ▾
- Edit Data Source

**Navigation:**

- Connections 1
- Extract Refreshes 0
- Connected Workbooks 1
- Lineage** (selected)

**Fields (6):**

Type	Name	Sheets	Description	Sensitivity
Abc	Buying Group	2	No description	
Abc	Buying Office	1	No description	
#	Sales YoY	2	No description	
#	Tableau (Count)	0	No description	
#	VA YoY	1	No description	
#	Week	1	No description	

Filter by field name: enter search ▾ Sort By: Name (a-z) ▾

**Lineage:**

- Databases 1
- Tables and Objects 1
- Tableau (Tableau practice) Fields 6** (Selected)
- Workbooks 1
- Sheets 2
- Dashboards 1
- Owners 1

# Tableau lineage on steroids

```
query DatasourceDetails($projectName: String!){  
  publishedDatasources(filter:{projectName: $projectName} ){  
    luid  
    name  
    downstreamWorkbooks{  
      luid  
      name  
    }  
  }  
}
```

See: [Tableau Metadata API](#), [GraphQL](#)

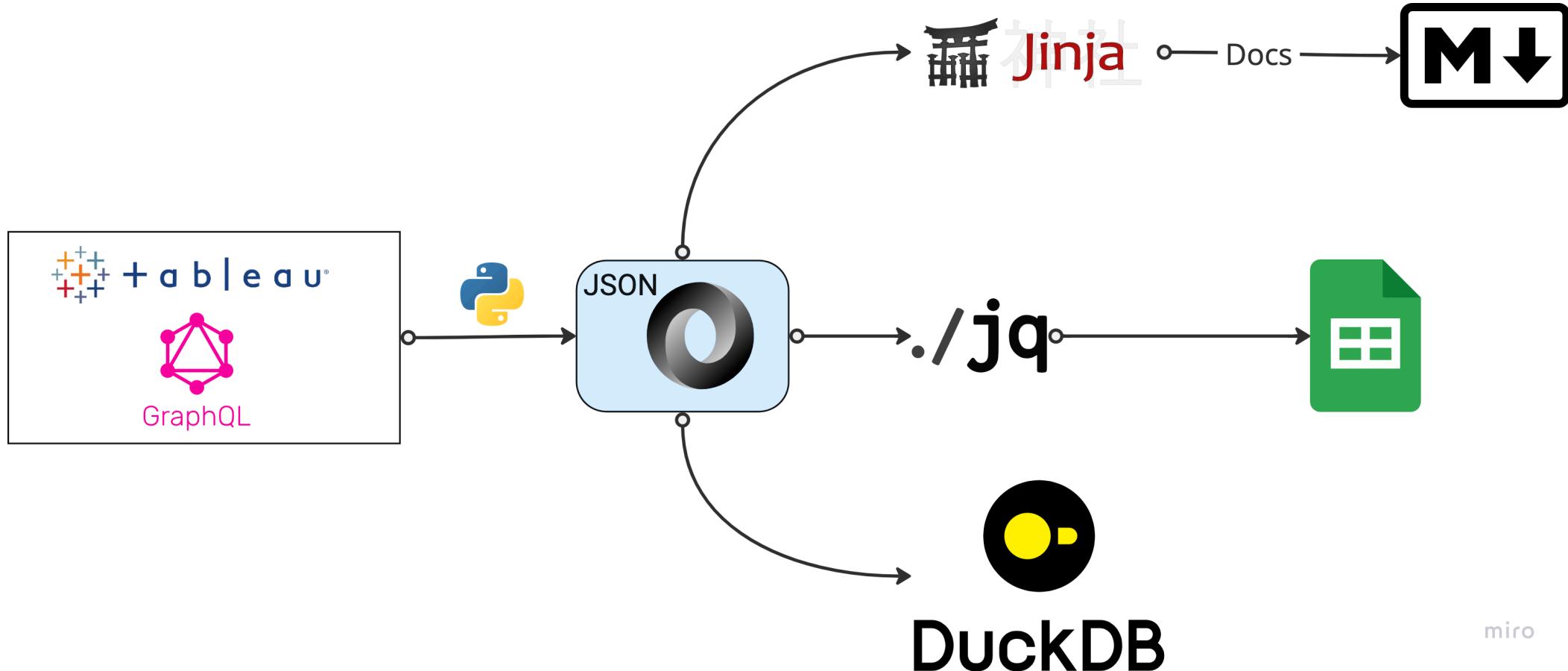
# Just enough Python

```
with tableau_server.auth.sign_in(tableau_auth):
    variables = {"projectName": os.environ.get('PROJECT_NAME')}
    query = open('resources/graphql/simple_lineage.graphql').read()

    ## This is the important bit below
    query_results = tableau_server.metadata.query(query, variables=variables)
    with open('simple_lineage.json', 'w') as f:
        json.dump(query_results, f)
```

See: [Tableau Server Python Client](#)

# (Meta)Data Engineering



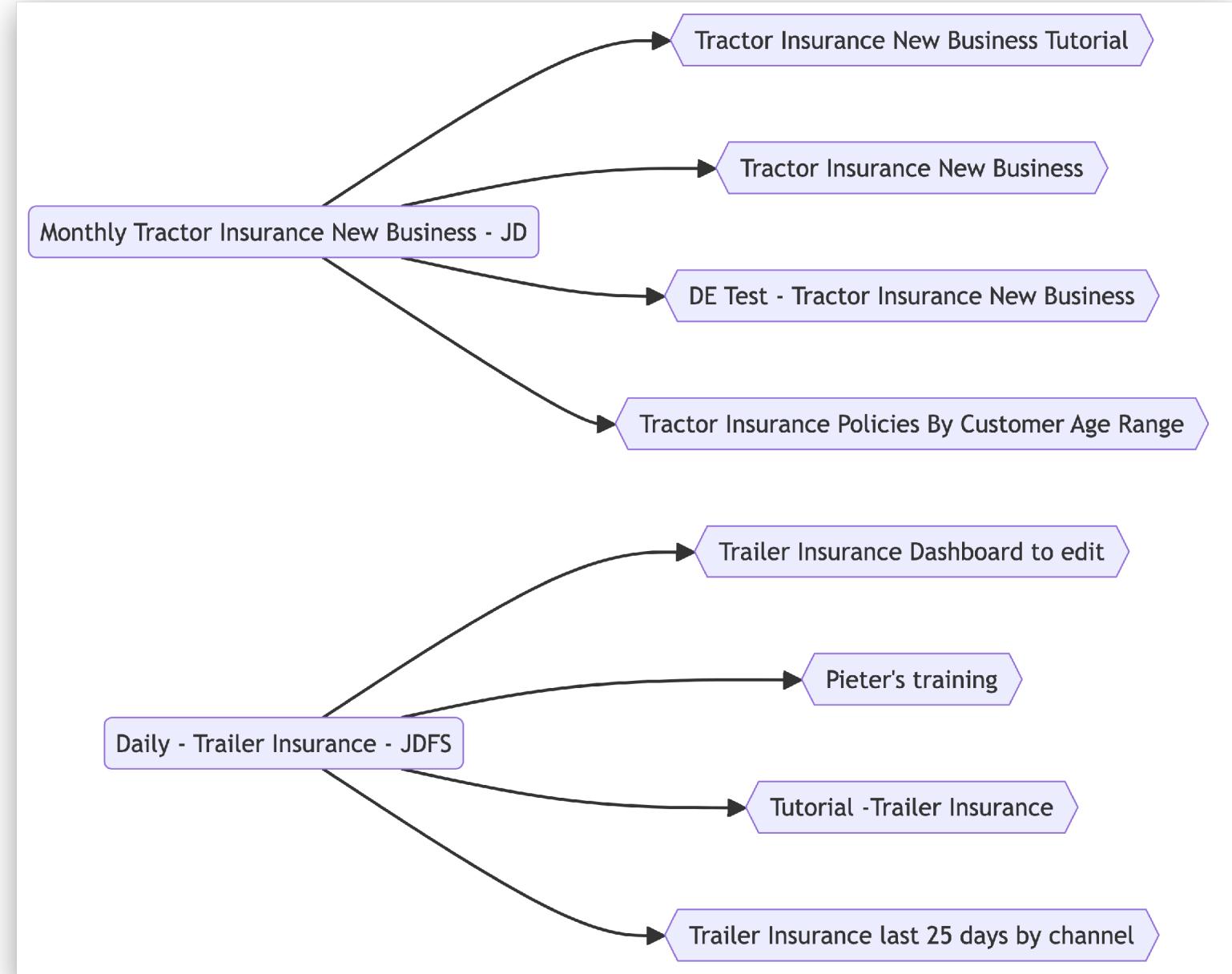
miro

# Jinja template

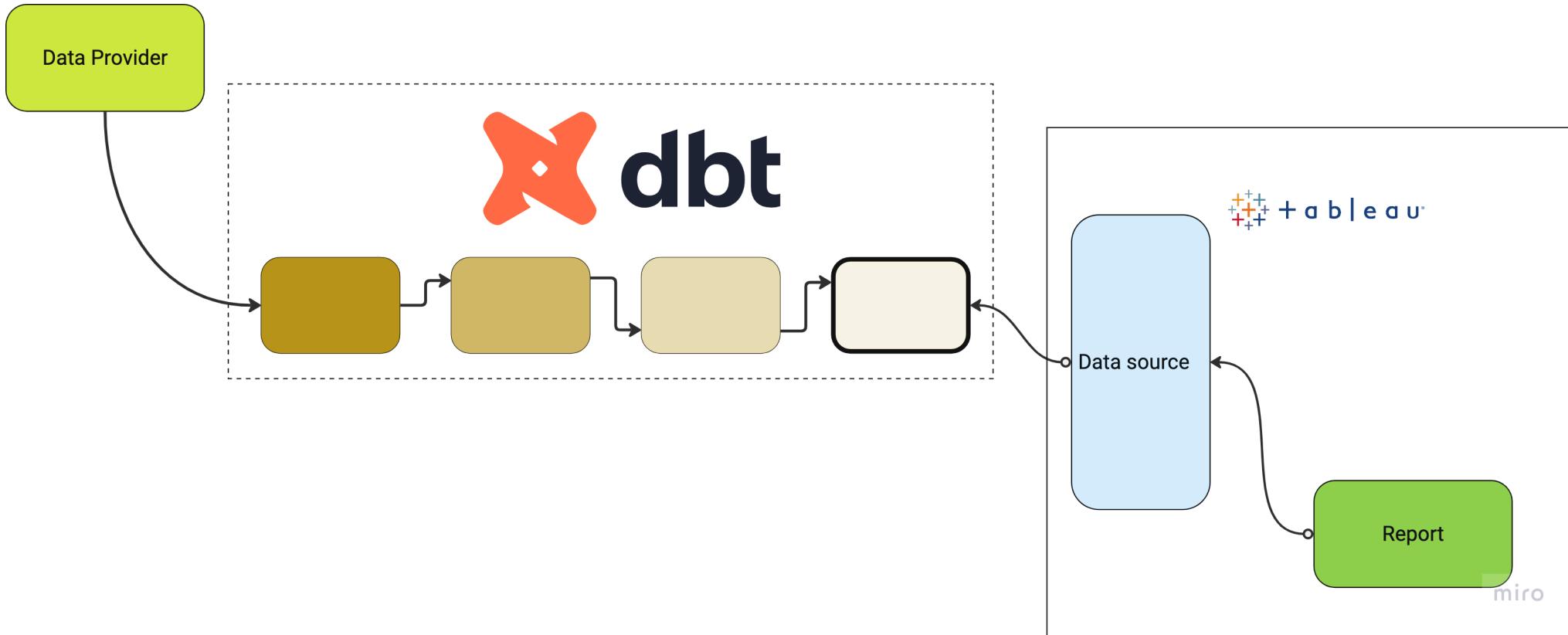
```
flowchart LR
{% for ds in data.publishedDatasources %}
    {{ ds.luid }}({{ ds.name }})
    {% for wb in ds.downstreamWorkbooks %}
        {% if wb.name != None %}
            {{ wb.luid }}["{{ wb.name }}"]
            {{ ds.luid }} --> {{ wb.luid }}
        {% endif %}
    {% endfor %}
{% endfor %}
```

See: [Jinja](#), [MermaidJS flowcharts](#)

# The little mermaid



# Extend right: data pipelines with dbt



# When Business complains 😠

The figures are incorrect!

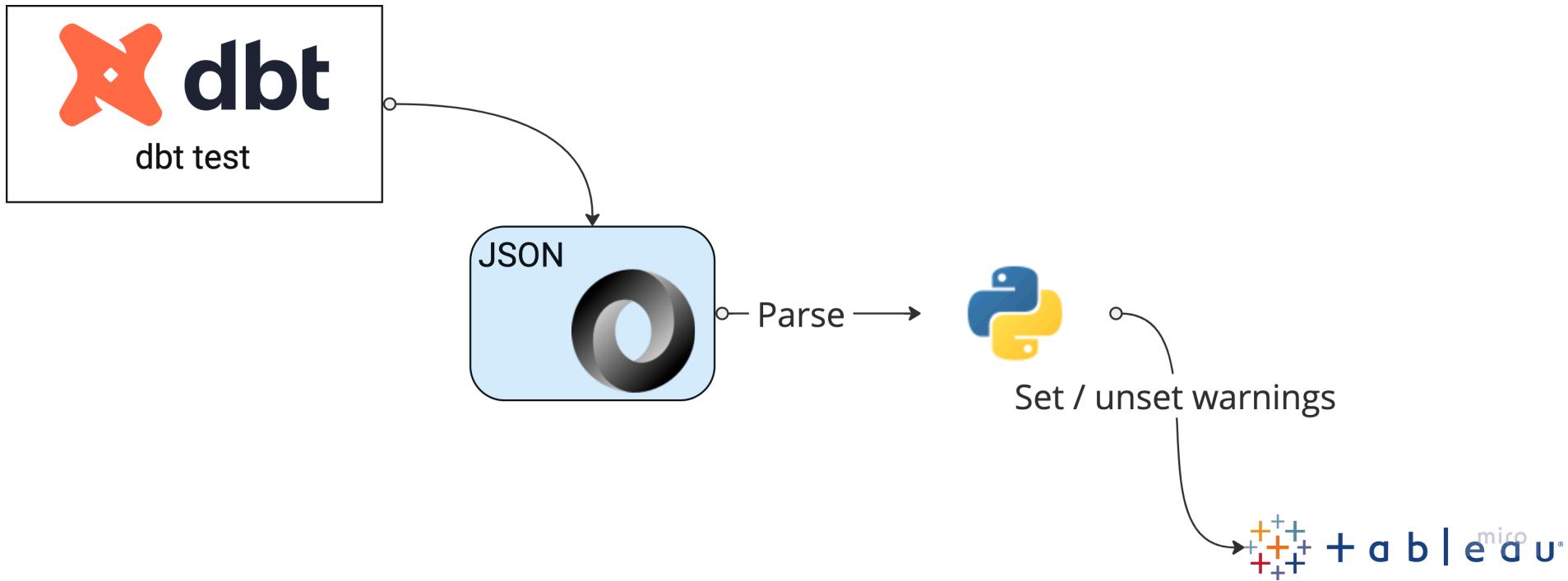
Data could be late?

Data could be wrong?

# Operability

- Data quality checks (e.g. dbt tests)
- Slack alerts on failures
- Not so business friendly 😢

# From checks to labels



# Sample dbt code

```
---
```

```
models:
  - name: tractor_sales
    columns:
      - name: created_at
        tests:
          - dbt_expectations.expect_row_values_to_have_recent_data:
              datepart: day
              interval: 1
              tags: ['tableau_recency']
```

See: [dbt expectations](#)

# Just enough Python

```
table_item: TableItem = tableau_server.tables.get_by_id(luid)

data_quality_warning: DQWItem = DQWItem(
    warning_type=DQWItem.WarningType.WARNING,
    message="This asset (Table/View) has stale data.",
    active=True,
    severe=True)

tableau_server.tables.add_dqw(table_item, data_quality_warning)
```

# What does it look like?

The screenshot shows a Tableau dashboard titled "Tableau (Tableau practice)". The dashboard interface includes a sidebar with navigation links such as Home, Favourites, Recents, Shared with Me, Recommendations, Personal Space, Collections, Explore (which is selected), and External Assets. The main workspace displays a list of fields under "Fields (6)". A modal window titled "Warning" is open, stating: "This asset (Table/View) has stale data. Set by Financial Services 23 Aug 2024, 10:44". The "Lineage" section on the right shows the hierarchy of the asset, starting from "Tableau (Tableau practice)" which contains 6 fields, leading down to 1 workbook, 2 sheets, 1 dashboard, and 1 owner.

Type	Name	Sheets	Description
Abc	Buying Group	2	No description
Abc	Buying Office	1	No description
#	Sales YoY	2	No description
#	Tableau (Count)	0	No description
#	VA YoY	1	No description
#	Week	1	No description

## Shift left: should you push this to production?

```
SELECT applicationId AS application_id,  
       datecreated as DATE_CREATED  
  from transactions;
```

# SQLFluff says NO!!!

```
== [/Users/guillaumebelrose/Work/Presentations/dataconf24/dataconf-demo/resources/sql/lint.sql] FAIL
L:  1 | P:  1 | LT09 | Select targets should be on a new line unless there is
| only one select target. [layout.select_targets]
L:  1 | P:  7 | LT02 | Expected line break and indent of 4 spaces before
| 'applicationId'. [layout.indent]
L:  1 | P:  8 | CP02 | Unquoted identifiers must be consistently upper case.
| [capitalisation.identifiers]
L:  2 | P: 17 | CP01 | Keywords must be consistently upper case.
| [capitalisation.keywords]
L:  7 | P:  1 | LT12 | Files must end with a single trailing newline.
| [layout.end_of_file]
All Finished 🎉 !
```

# Keeping reporting assets consistent

- Naming
  - Names | Descriptions
  - Tags
- Configuration
  - Caching
  - Database connection settings
- **Automated checks** over written guidelines

# Roll Your Own Linter



## Shift left: The effects of UI-driven work

Time consuming to create | re-create

Time consuming to verify

Lack of audit trail

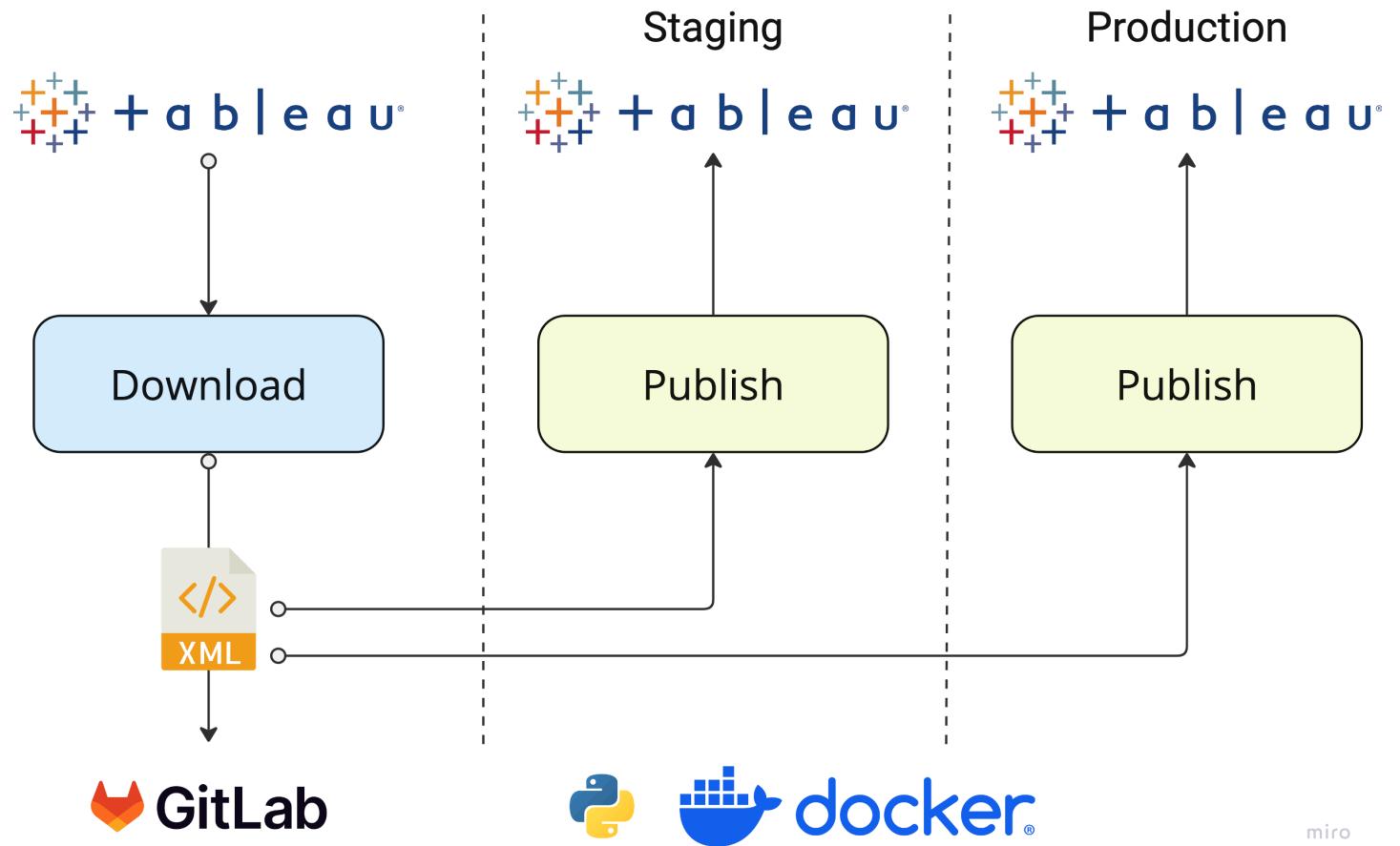
## Lack of release pipeline

Less appreciation for segregated environments

Dev work becomes productionised

Self-serve vs dependable data

# Data Source As Code



miro

# What it enables

**Improved quality: dev -> qa -> prod**

**Common deployment patterns for reporting and data pipelines**

**Audit trail**

# To recap

## Tales from production

- Applied this thinking to our domain
- Easier to adopt for green field projects
- Many areas are ripe for automation
- **Ways of working** just as crucial

# Thank you!

To my awesome team

...and the awesome audience 😊

PS: don't ask me about PowerBI