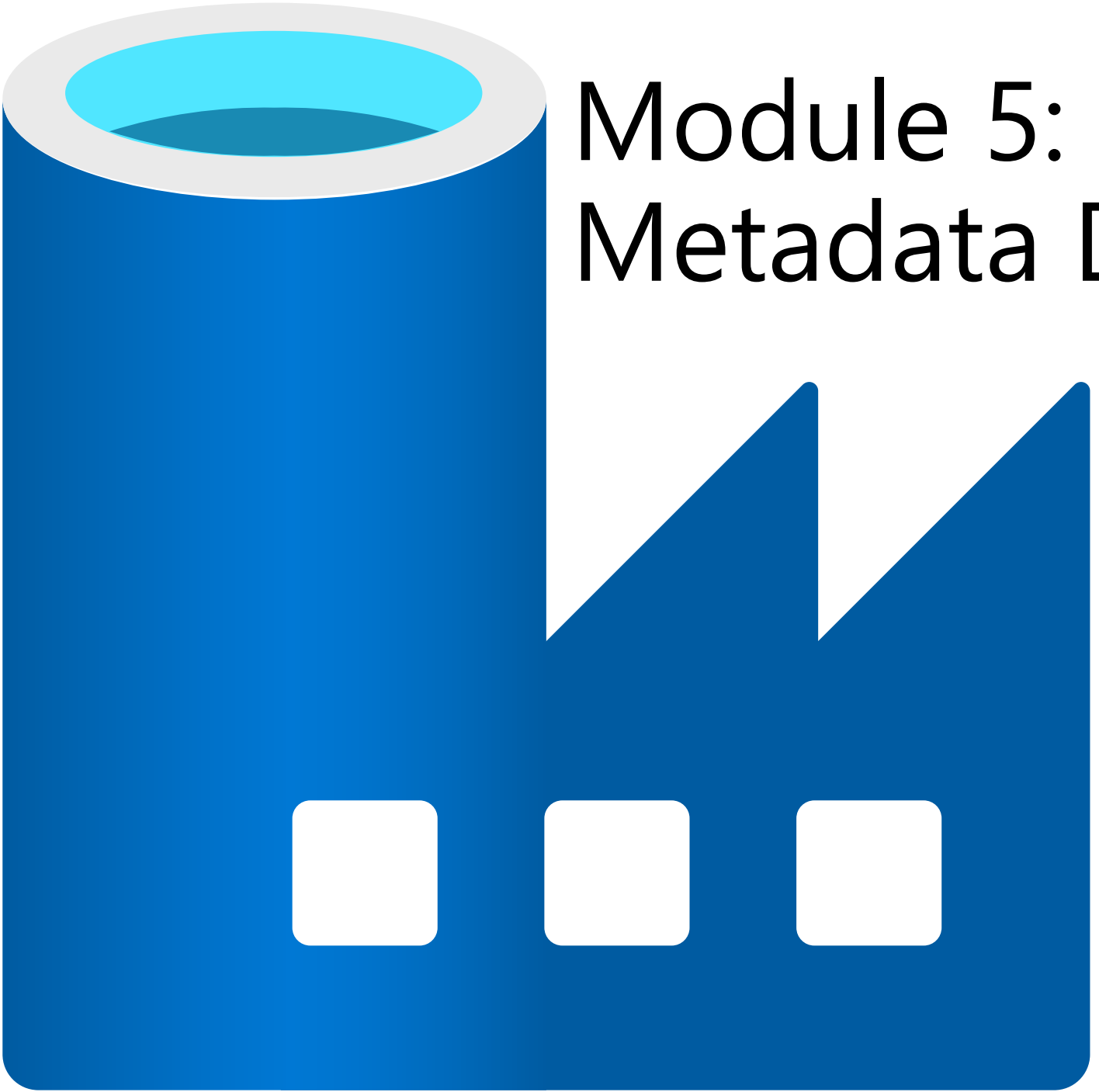
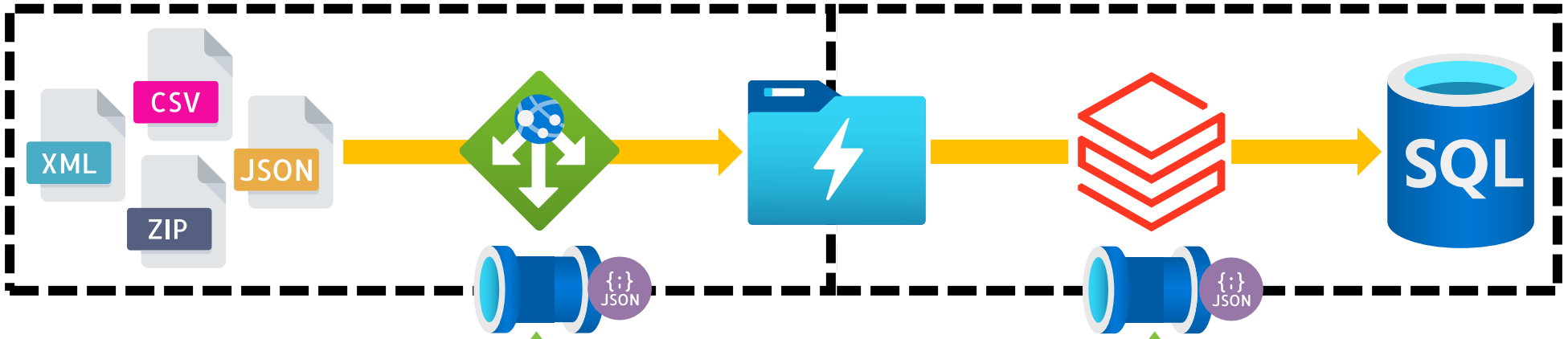


Module 5:
Metadata Driven Pipelines

- Expressions
- Dynamic Pipeline
- Orchestration Framework
  - procfwk.com

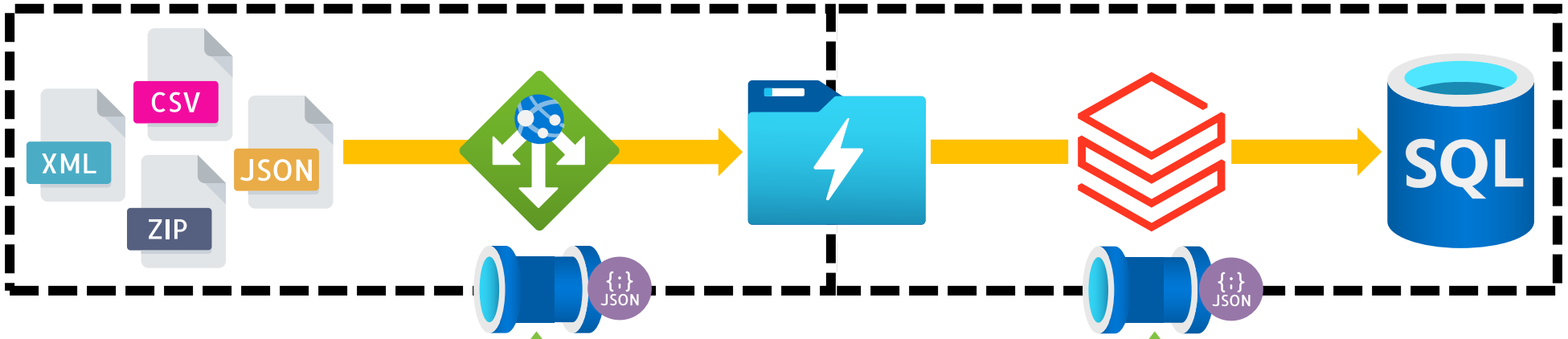
Expressions

Data Factory Components

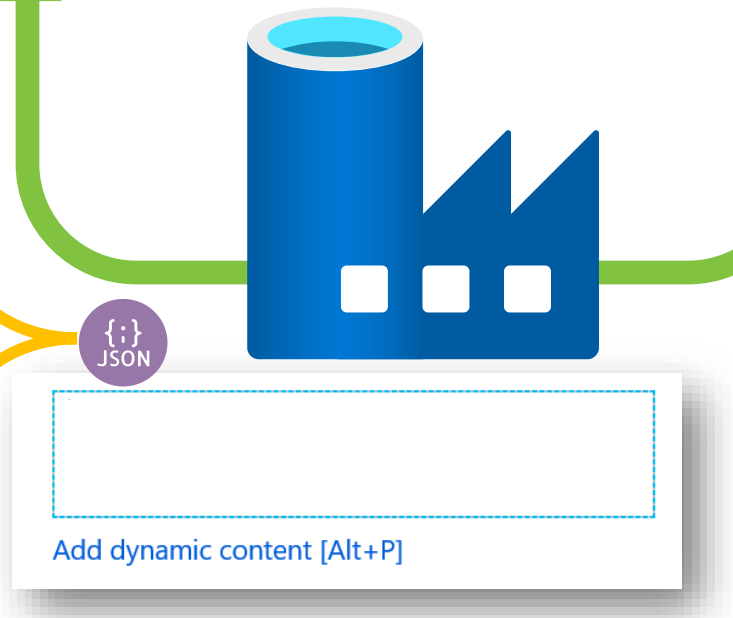


1. **Linked Services**
2. **Datasets**
3. **Activities**
4. **Pipelines**
5. **Triggers**

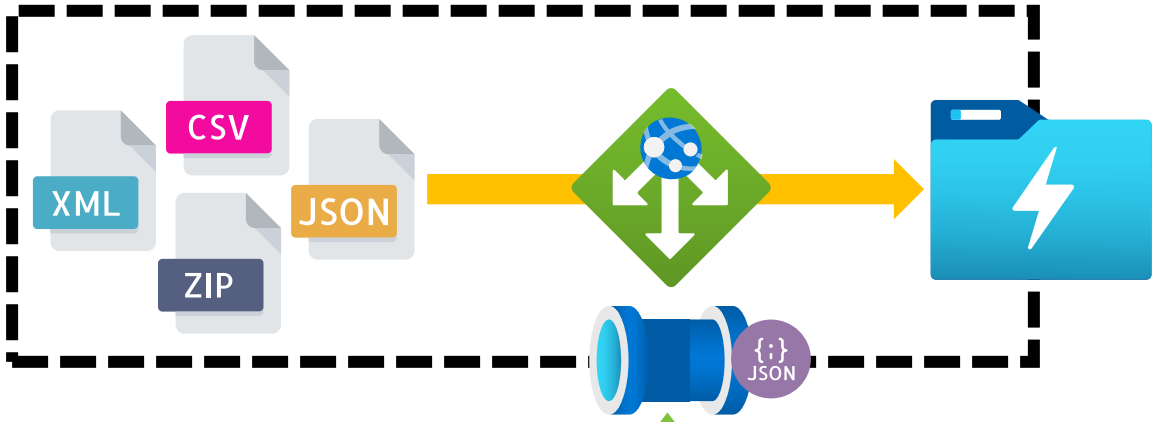
Data Factory Components – Add Dynamic Content



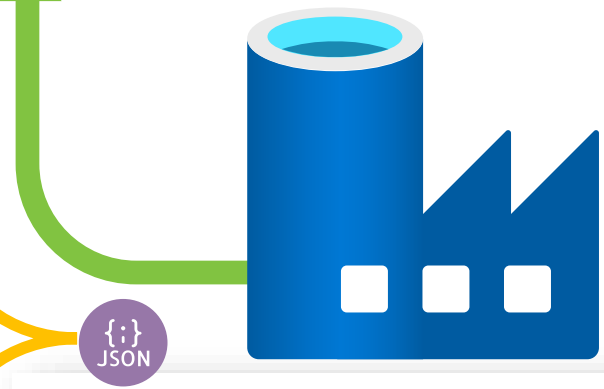
1. **Linked Services**
2. **Datasets**
3. **Activities**
4. **Pipelines**
5. **Triggers**



Control Flow Expressions



1. **Linked Services**
2. **Datasets**
3. **Activities**
4. **Pipelines**
5. **Triggers**



Add dynamic content

@pipeline().parameters.DemoParam

Clear contents

Filter...

Use expressions, functions or refer to system variables.

- System variables
  - Data factory name: Name of the data factory the pipeline run is running within
  - Pipeline Name: Name of the pipeline
  - Pipeline run ID: ID of the specific pipeline run
  - Pipeline trigger ID: ID of the trigger that invokes the pipeline
  - Pipeline trigger name: Name of the trigger that invokes the pipeline
  - Pipeline trigger time: Time when the trigger that invoked the pipeline. The trigger time is the actual fired time, not the sch...
  - Pipeline trigger type: Type of the trigger that invoked the pipeline (Manual, Scheduler)
- Functions
  - Expand all
  - Collection Functions
  - Conversion Functions
  - Date Functions
  - Logical Functions
  - Math Functions
  - String Functions

https://docs.microsoft.com/en-us/azure/data-factory/control-flow-expression-language-

Control Flow Expressions

Save | Save as template | Validate | Debug

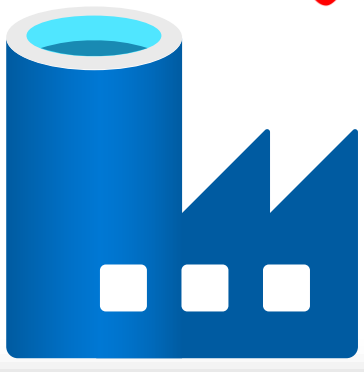
```
"SomethingDynamic": {
    "value": "@pipeline().parameters.DemoParam",
    "type": "Expression"
}
```

```
"SomethingDynamic": "@pipeline().parameters.DemoParam"
```

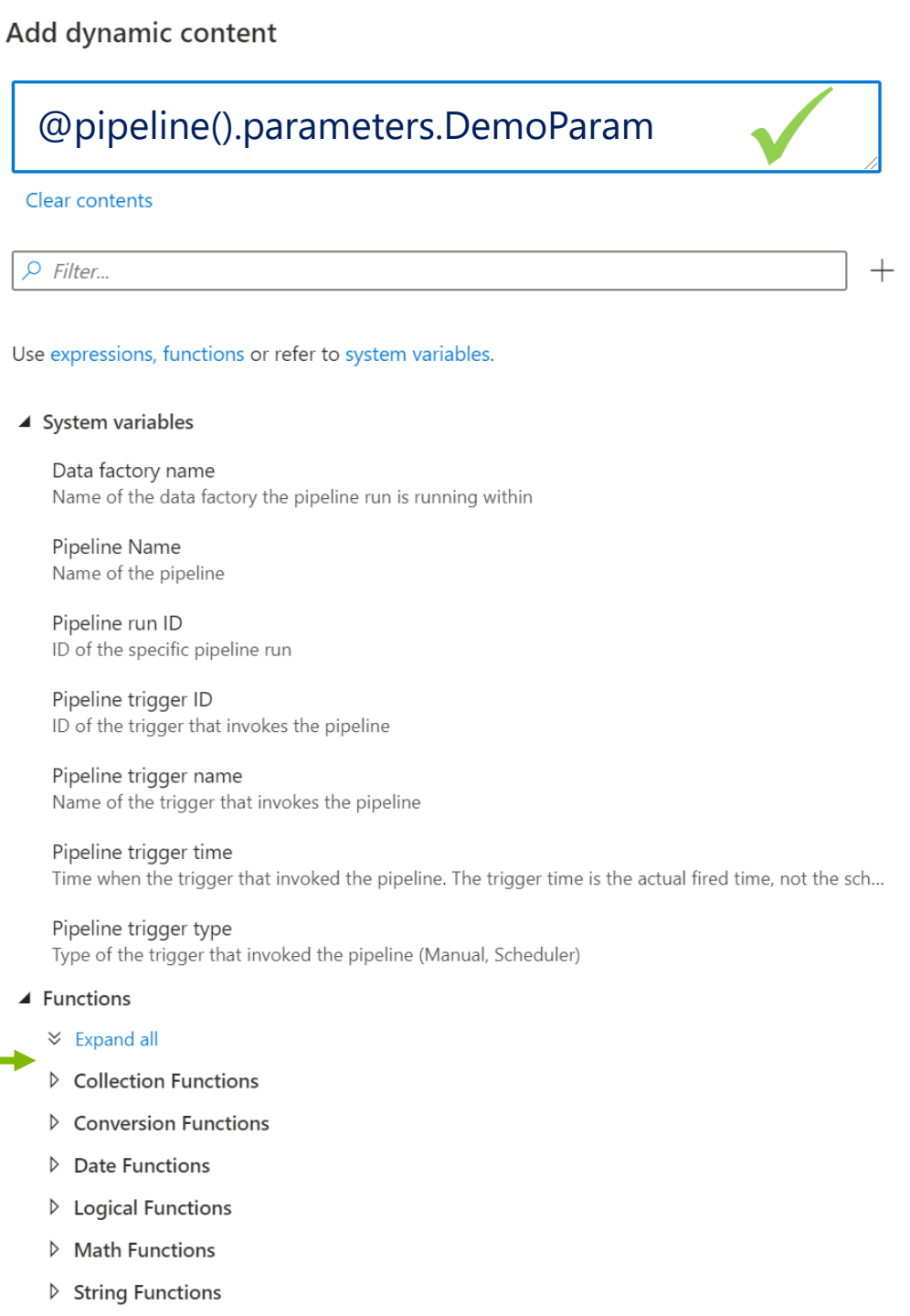
String Interpolation

```
"SomethingDynamic": "Hello World"
```

```
"parameters": {
   "DemoParam": {
   "type": "string",
   "defaultValue": "World"
}
```




```
"SomethingDynamic": "Hello @{pipeline().parameters.DemoParam}"
```

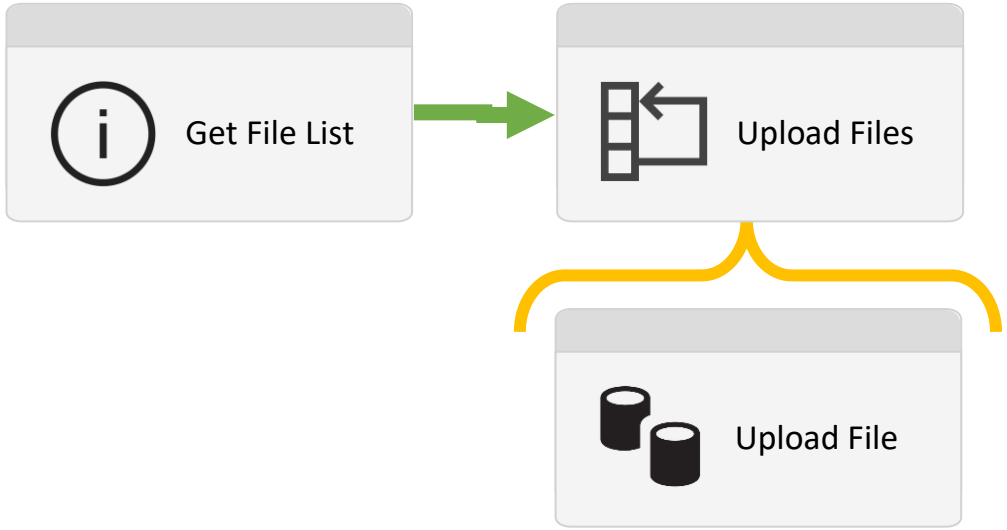


Dynamic Pipelines


DEMO

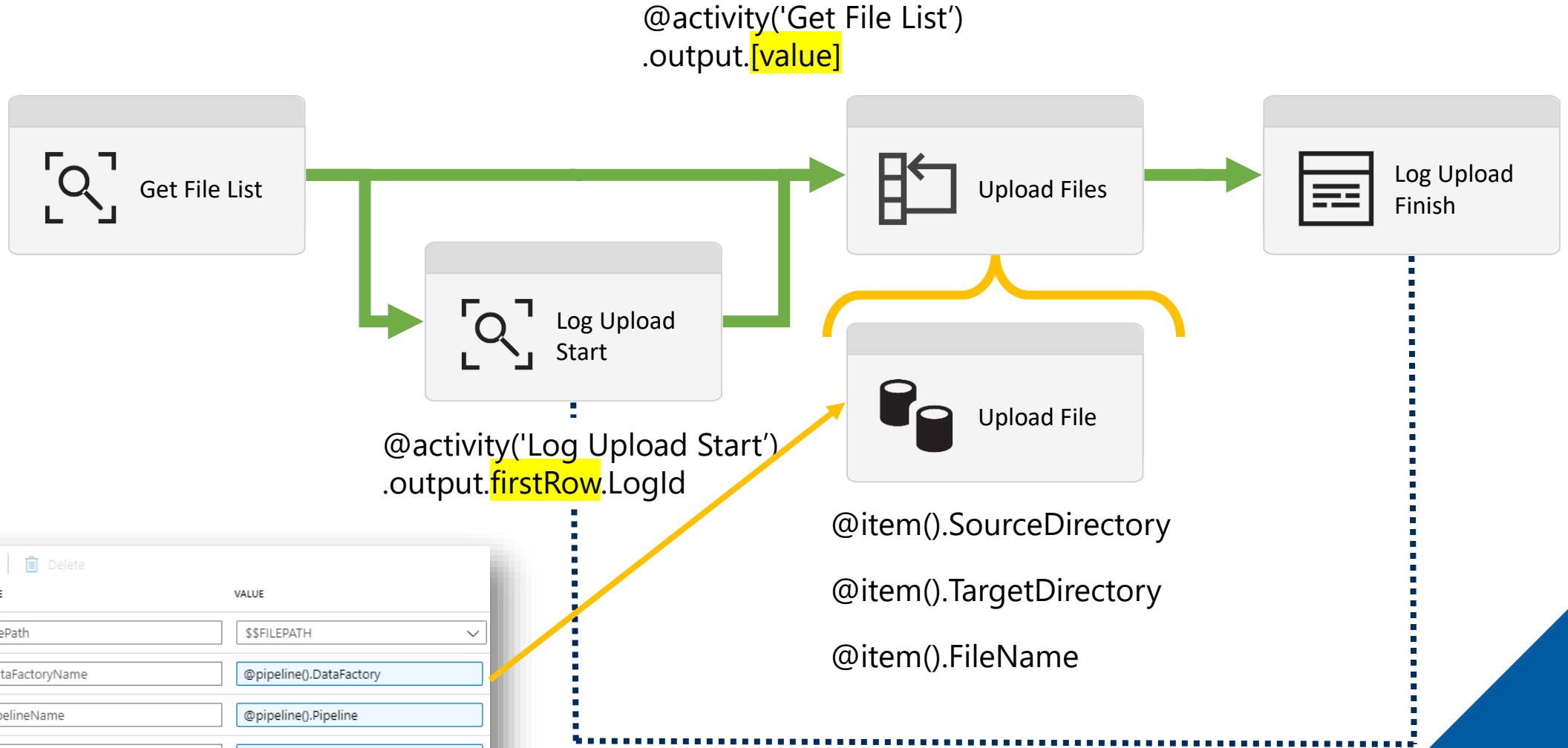
Demo Pipeline 1 – Discovery and Upload

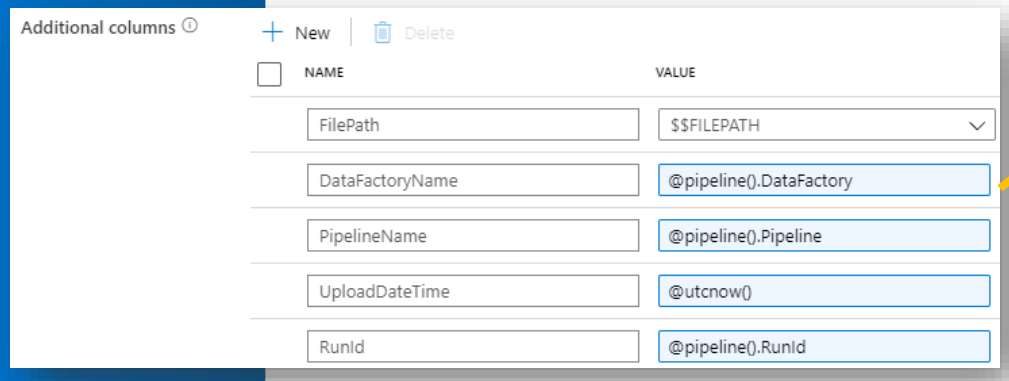
@activity('Get File List')
.output.[childItems]



@item().name

Demo Pipeline 2 – Simple Metadata and Upload

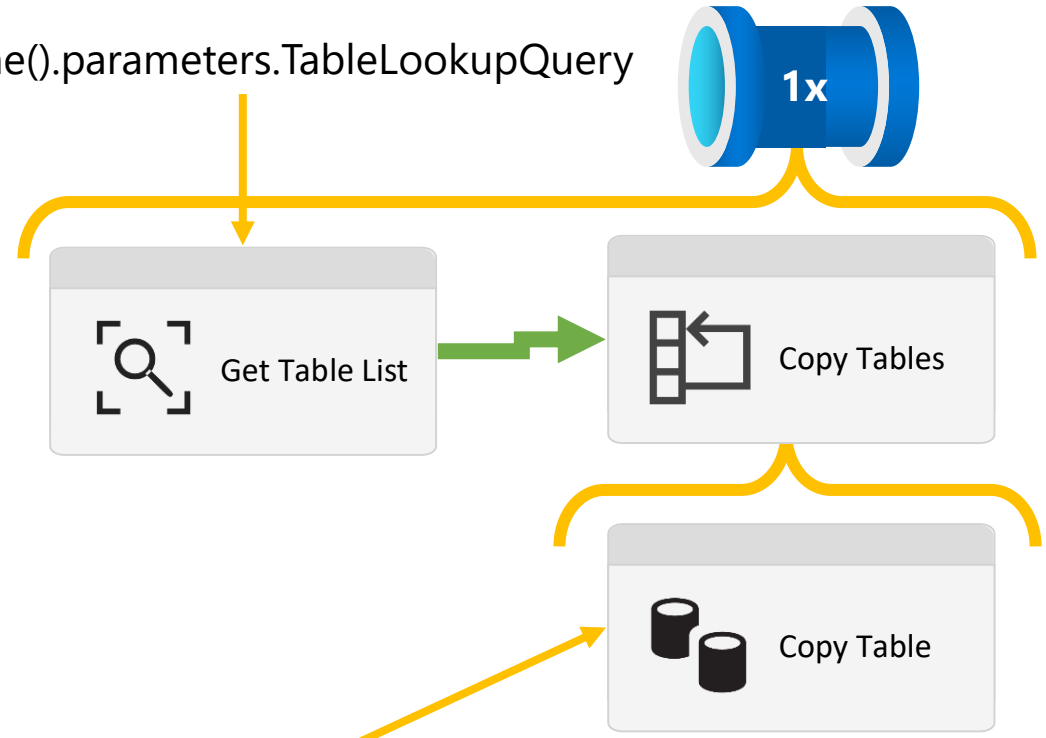


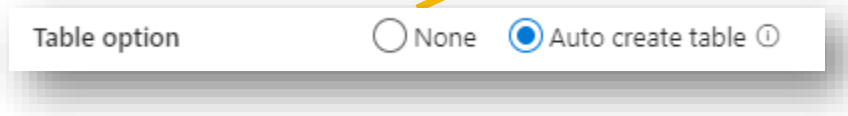


Demo Pipeline 3 – Lazy SQLDB Replication

```
SELECT s.name AS SchemaName, o.name AS TableName FROM sys.objects o
INNER JOIN sys.schemas s ON o.schema_id = s.schema_id WHERE o.[type] = 'U'
```

@pipeline().parameters.TableLookupQuery






```
IF OBJECT_ID(
                '@{item().SchemaName}.@{item().TableName}'
                ) IS NOT NULL

TRUNCATE TABLE @{item().SchemaName}.@{item().TableName}
```

1x

1x

@pipeline().parameters.SourceConnectionSecret

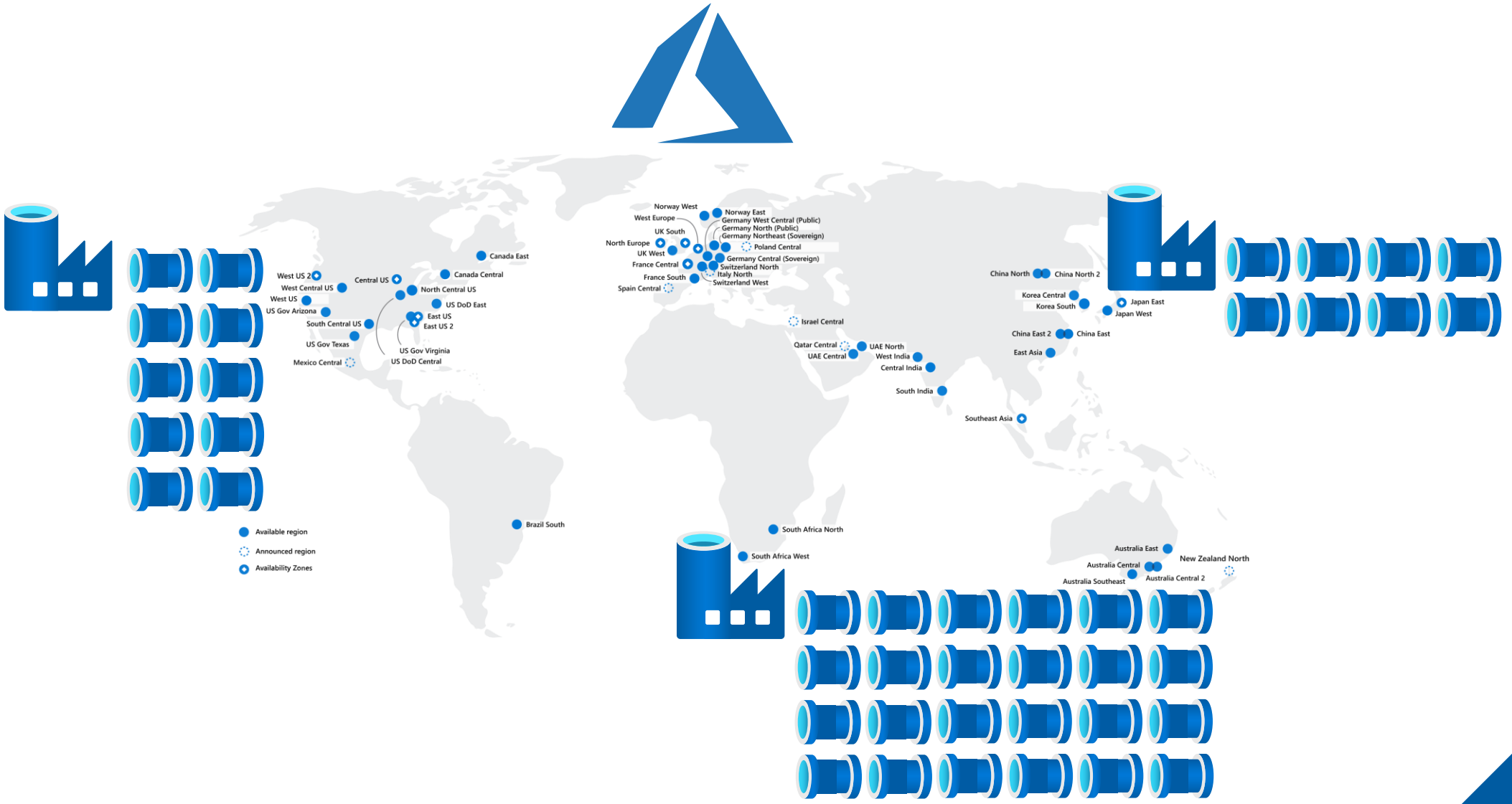
@pipeline().parameters.TargetConnectionSecret

@dataset().LinkedServiceConnectionSecret

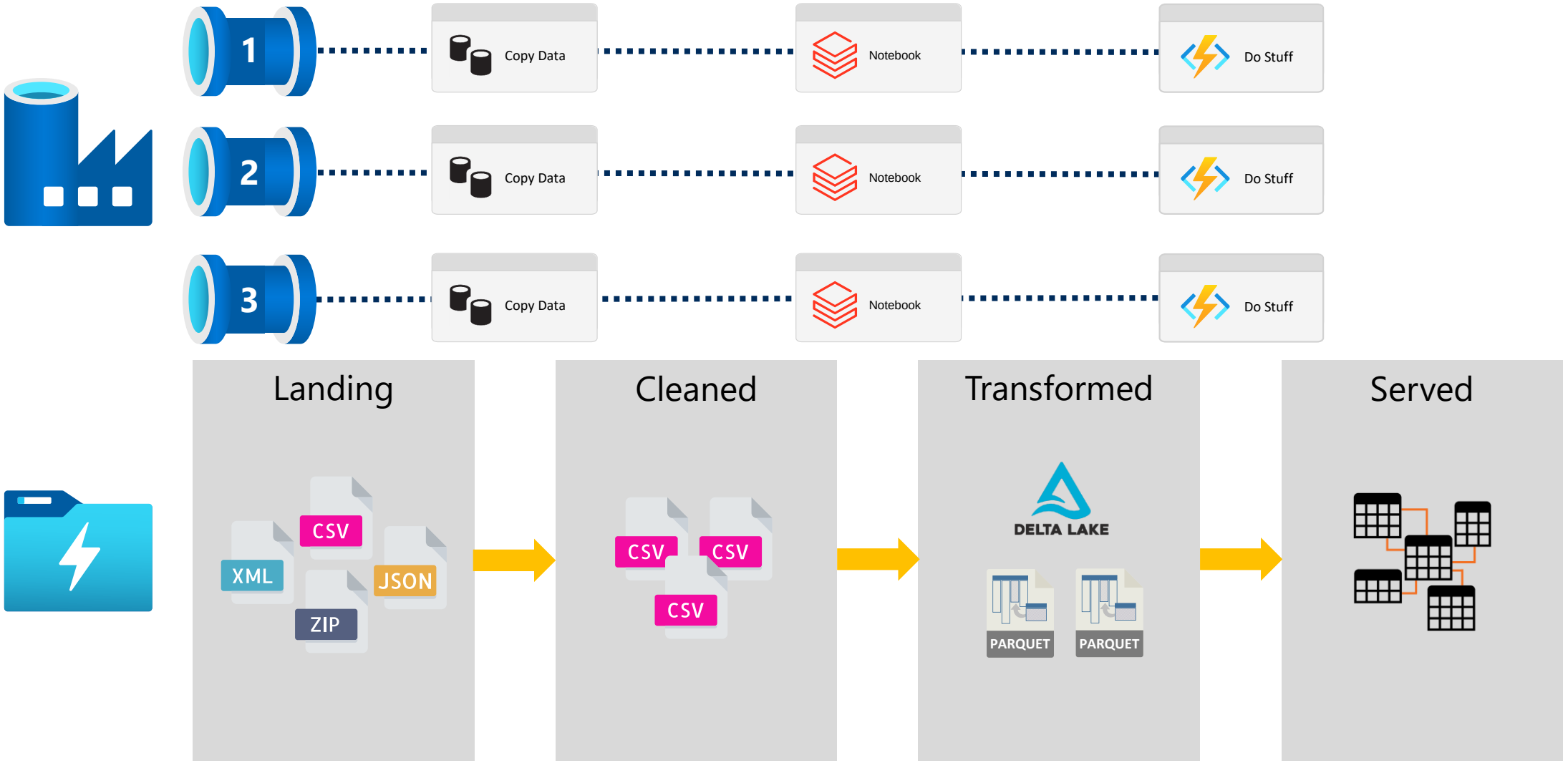
@linkedService().DBConnectionSecret

Orchestration Framework

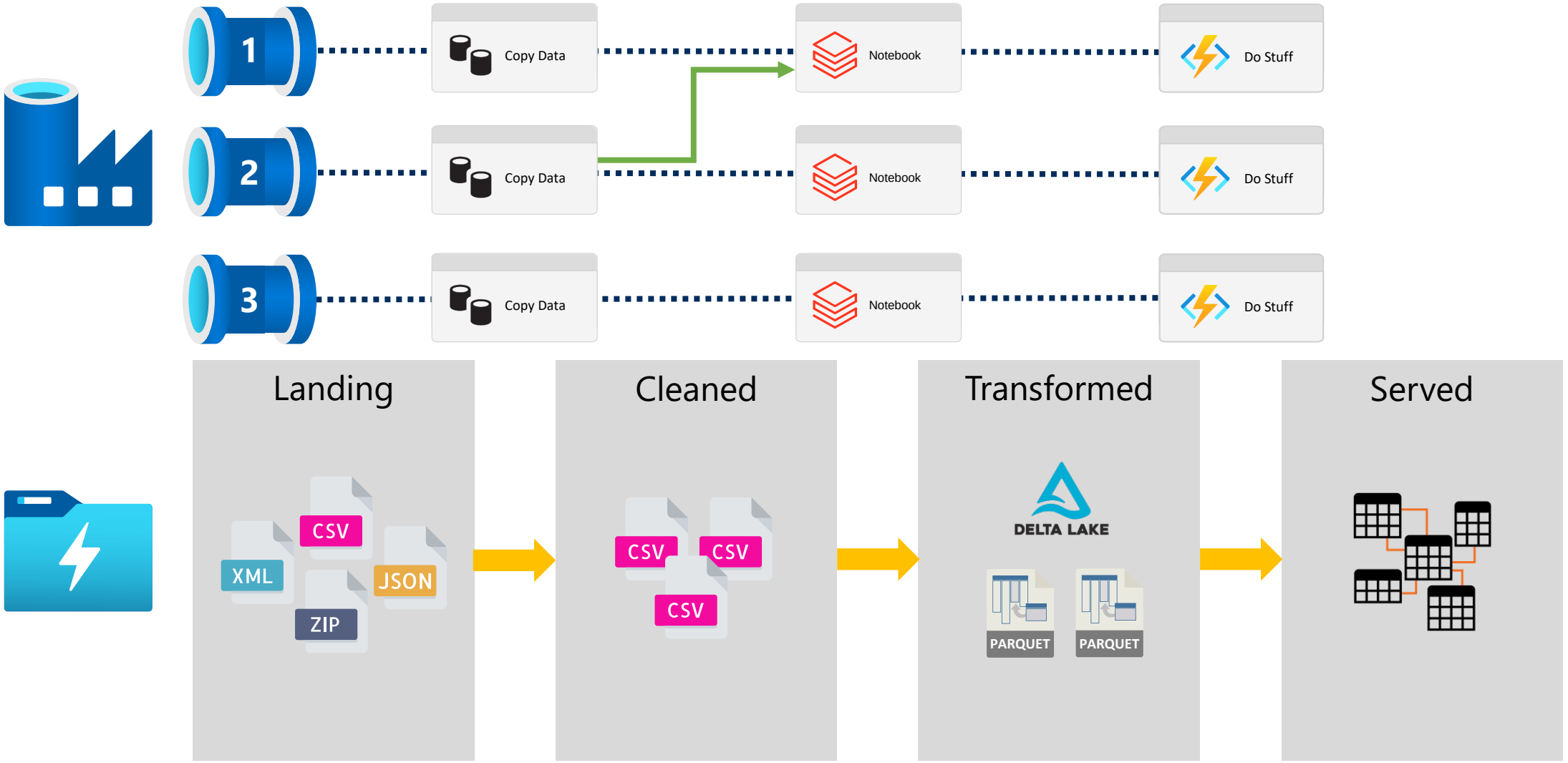
Problem: How should we structure our Data Factory Pipelines?



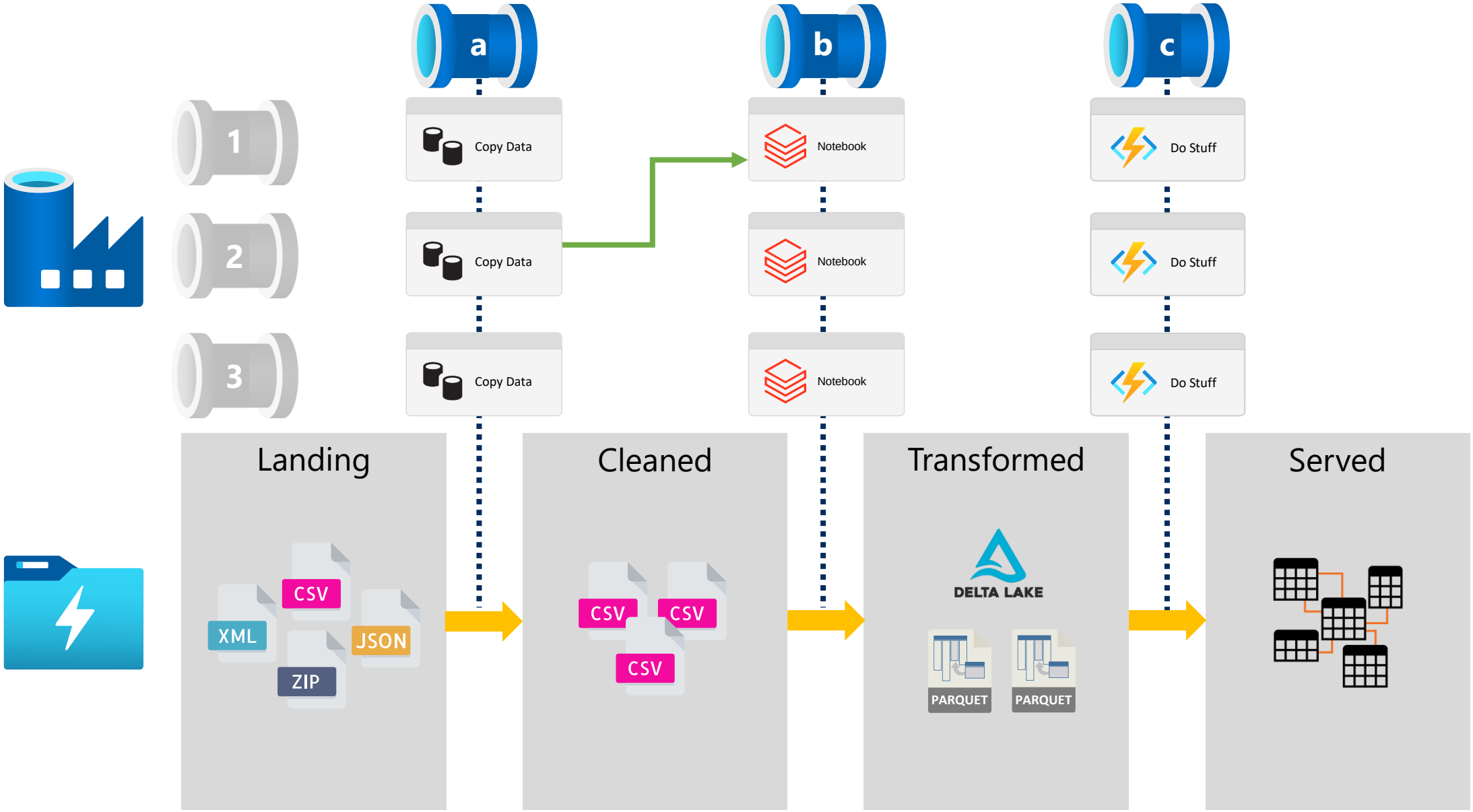
Problem


1
Copy Data
Notebook
Do Stuff
2
Copy Data
Notebook
Do Stuff
3
Copy Data
Notebook
Do Stuff
Landing
CSV
XML
JSON
ZIP
Cleaned
CSV
CSV
CSV
Transformed
DELTA LAKE
PARQUET
PARQUET
Served

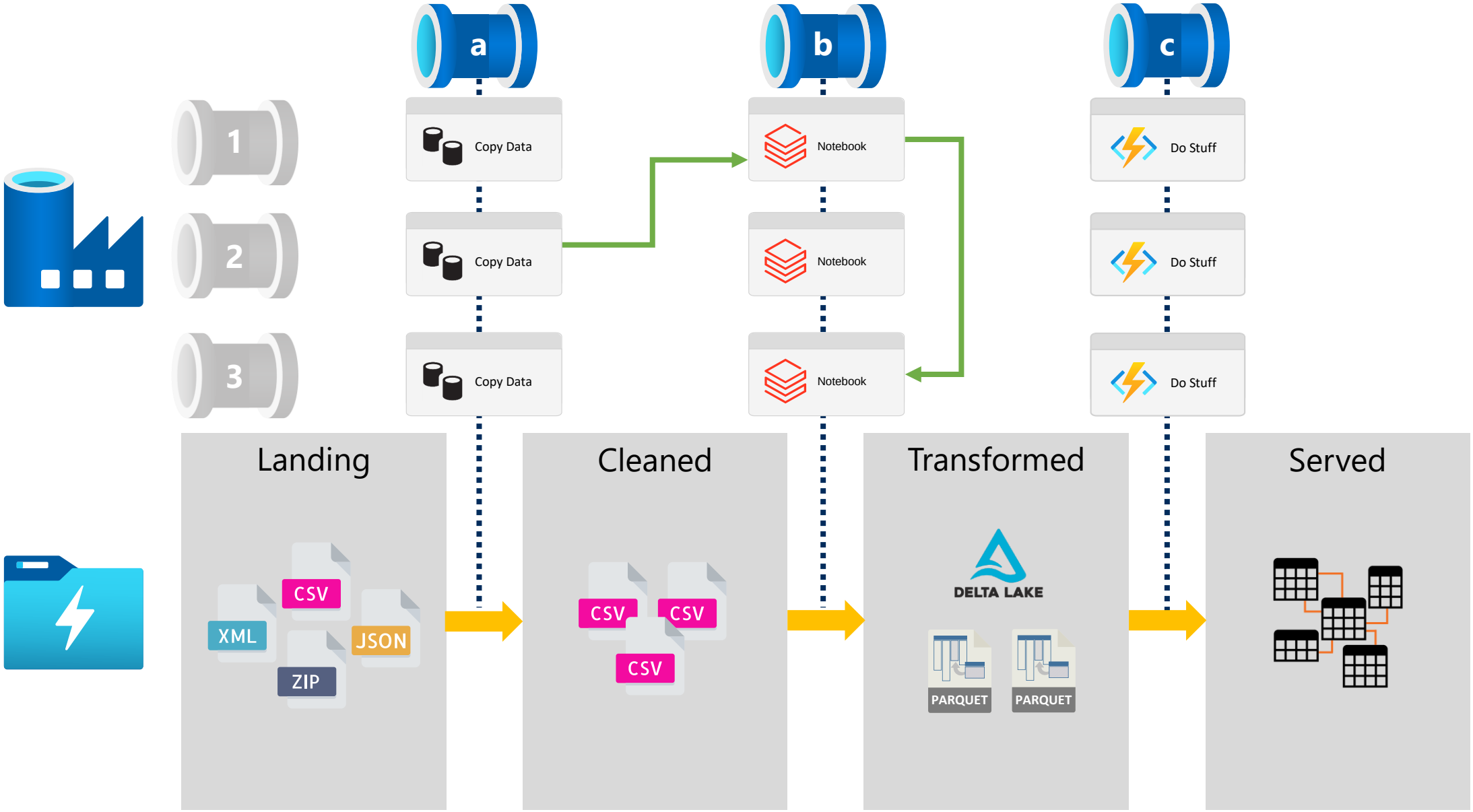
Problem


1
2
3
Copy Data
Copy Data
Copy Data
Notebook
Notebook
Notebook
Do Stuff
Do Stuff
Do Stuff
Landing
Cleaned
Transformed
Served
CSV
XML
JSON
ZIP
CSV
CSV
CSV
DELTA LAKE
PARQUET
PARQUET

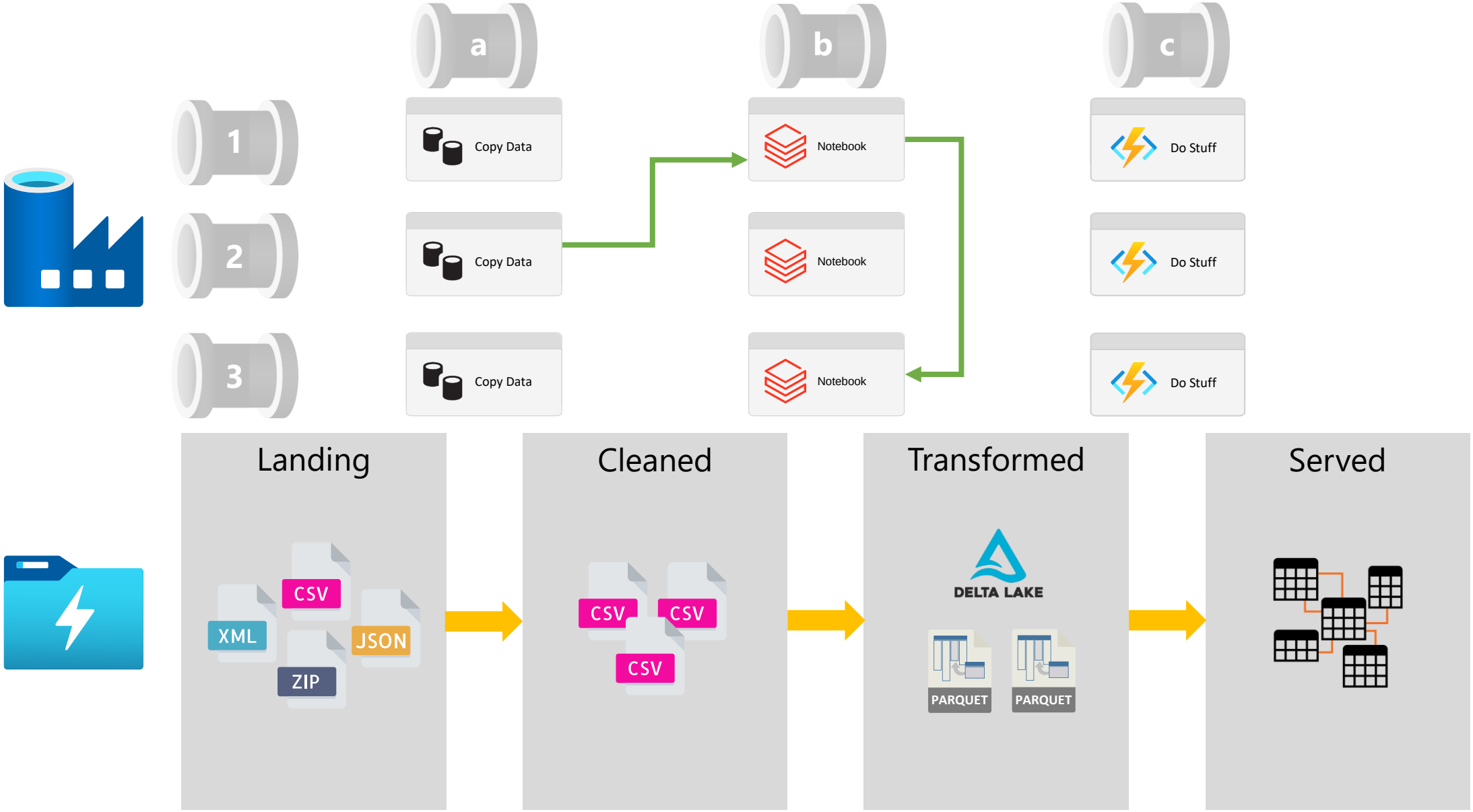
Problem


a
b
c
1
2
3
Copy Data
Copy Data
Copy Data
Notebook
Notebook
Notebook
Do Stuff
Do Stuff
Do Stuff
Landing
Cleaned
Transformed
Served
CSV
XML
JSON
ZIP
CSV
CSV
CSV
DELTA LAKE
PARQUET
PARQUET

Problem

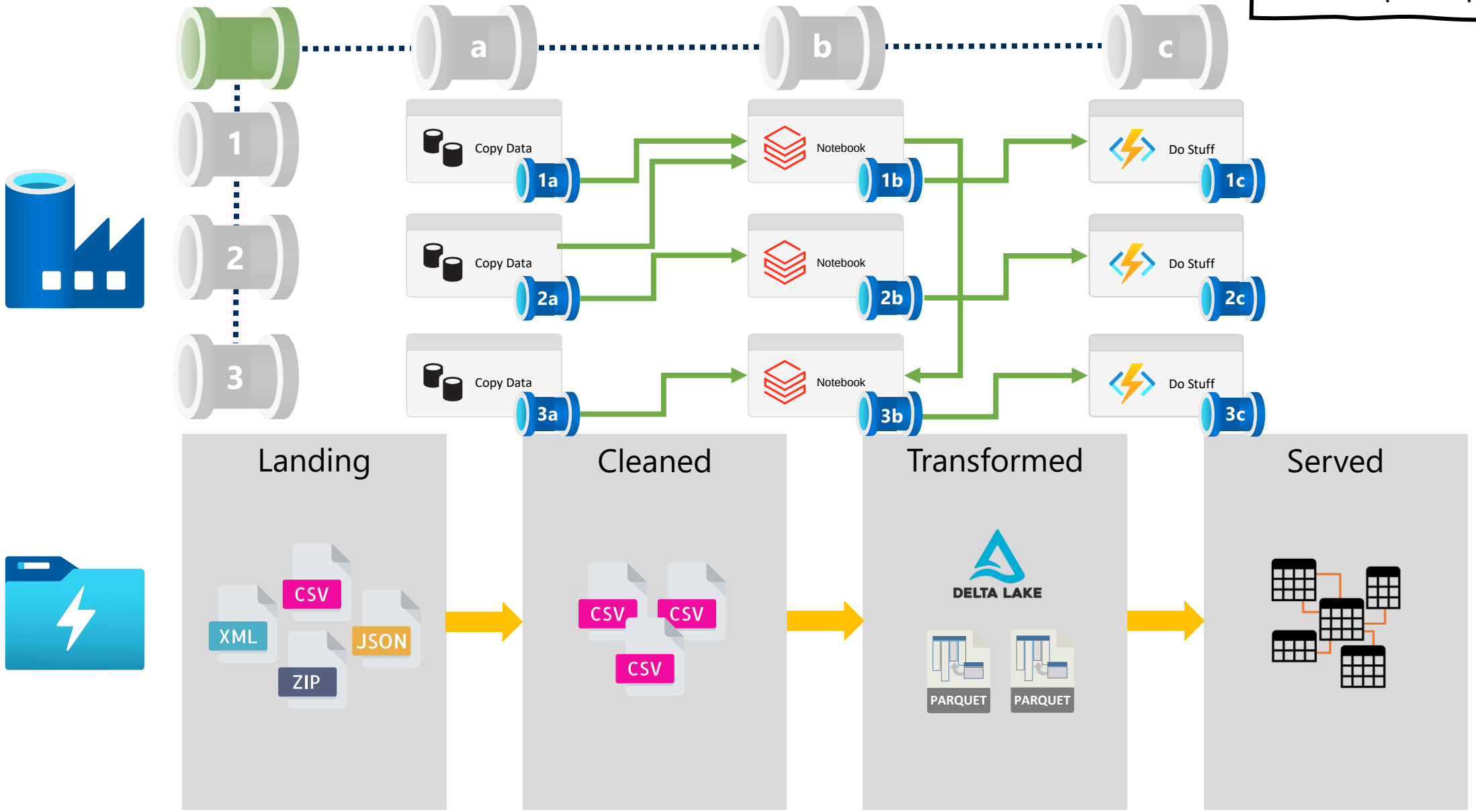

a
b
c
1
2
3
Copy Data
Copy Data
Copy Data
Notebook
Notebook
Notebook
Do Stuff
Do Stuff
Do Stuff
Landing
Cleaned
Transformed
Served
CSV
XML
JSON
ZIP
CSV
CSV
CSV
DELTA LAKE
PARQUET
PARQUET

Problem

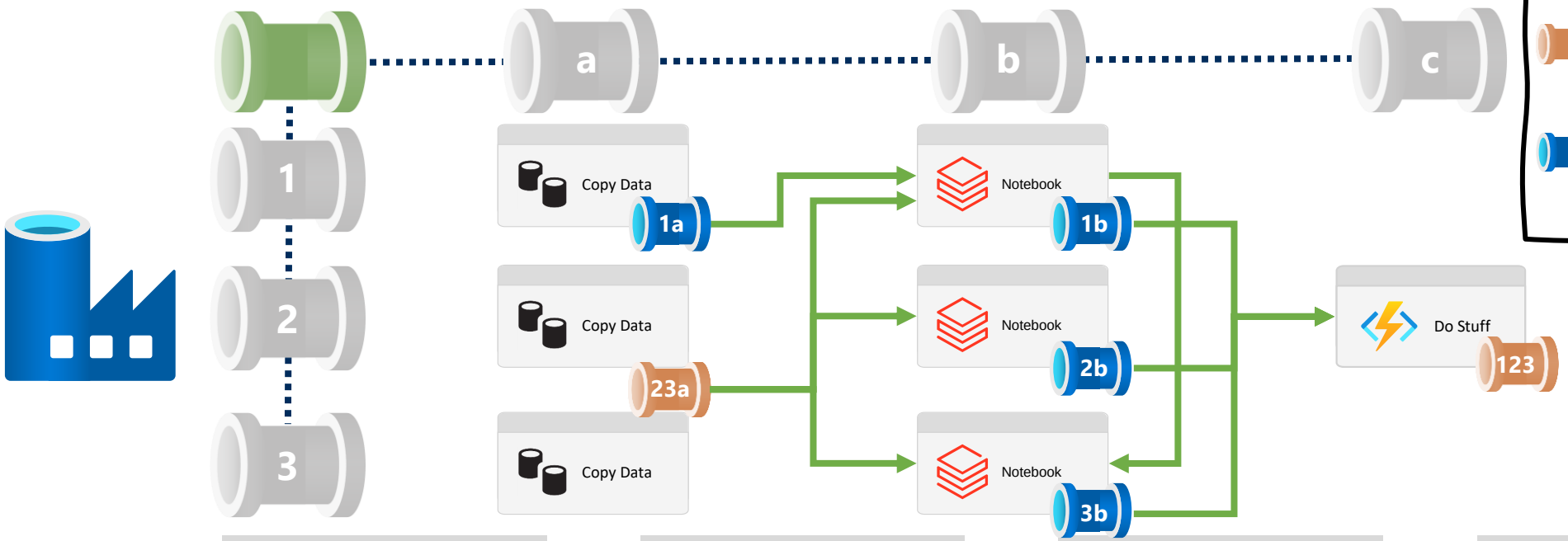

a
b
c
1
2
3
Copy Data
Copy Data
Copy Data
Notebook
Notebook
Notebook
Do Stuff
Do Stuff
Do Stuff
Landing
Cleaned
Transformed
Served
CSV
XML
JSON
ZIP
CSV
CSV
CSV
DELTA LAKE
PARQUET
PARQUET

Problem

Only 40 Activities per Pipeline.



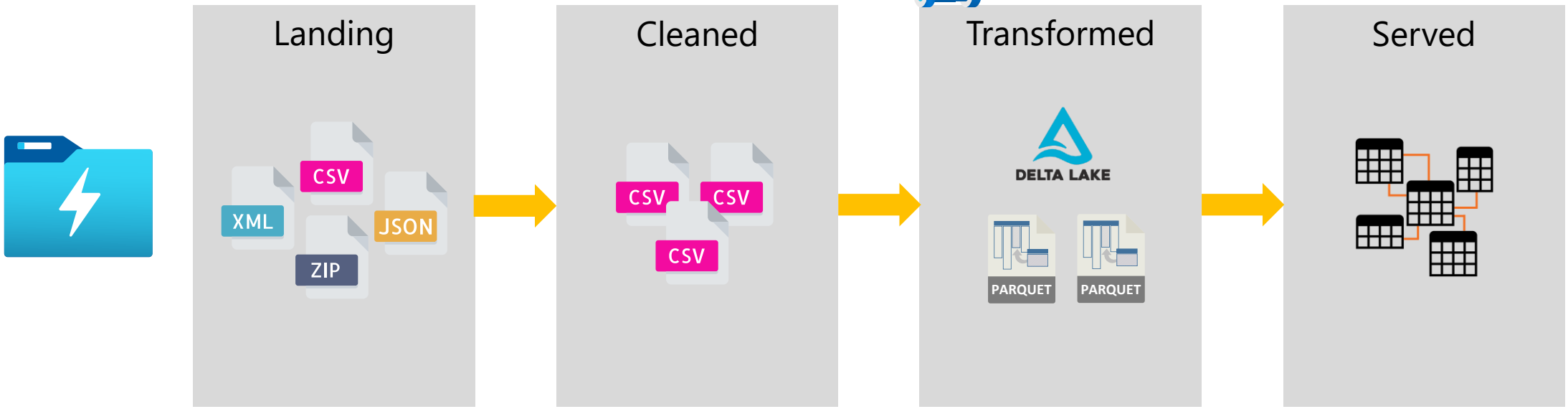
Problem



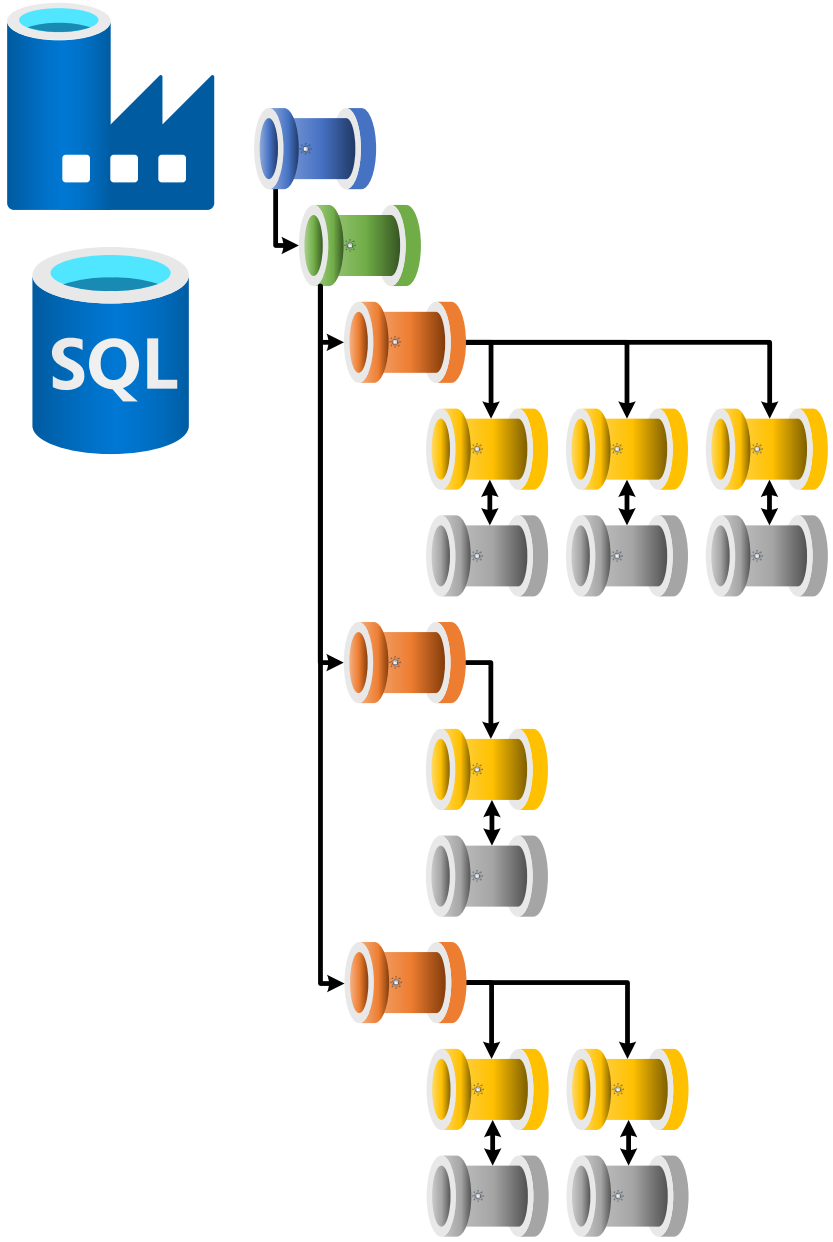
Grandparent pipeline for all processing.

Parent pipeline to consolidate work.

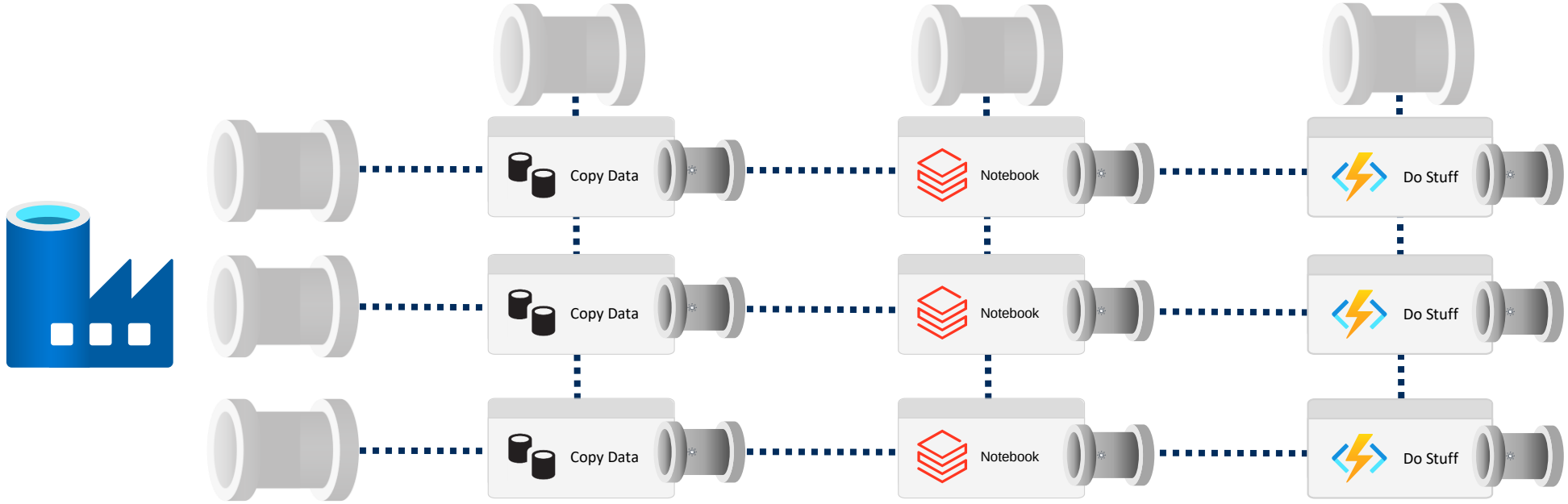
Child pipelines for low level dependencies.



Solution: Use Metadata to Drive Data Factory Pipelines



Solution

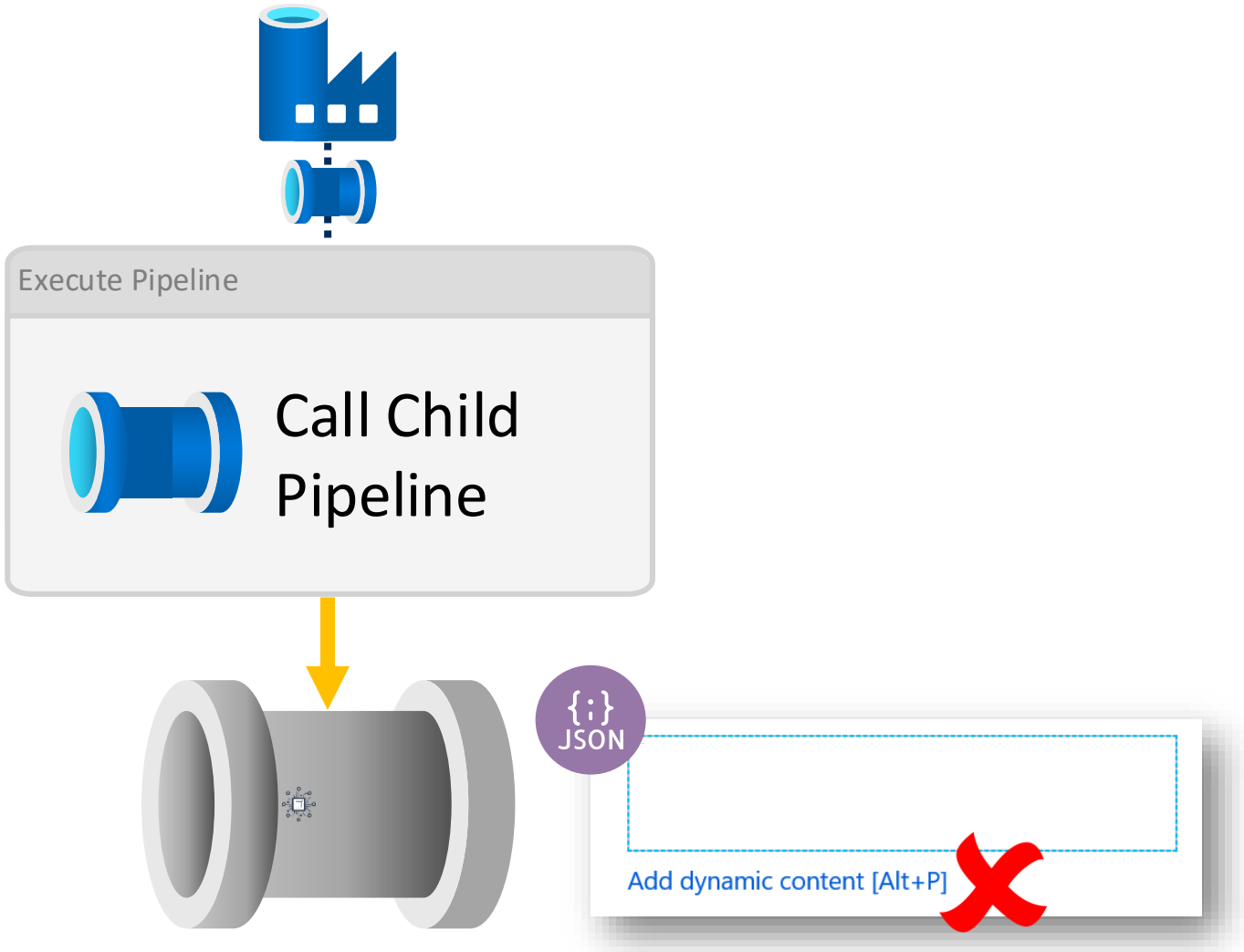


Stages
1
2
3

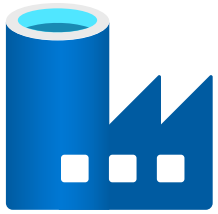
Pipelines
a
b
c
d
e
f
g
h
i

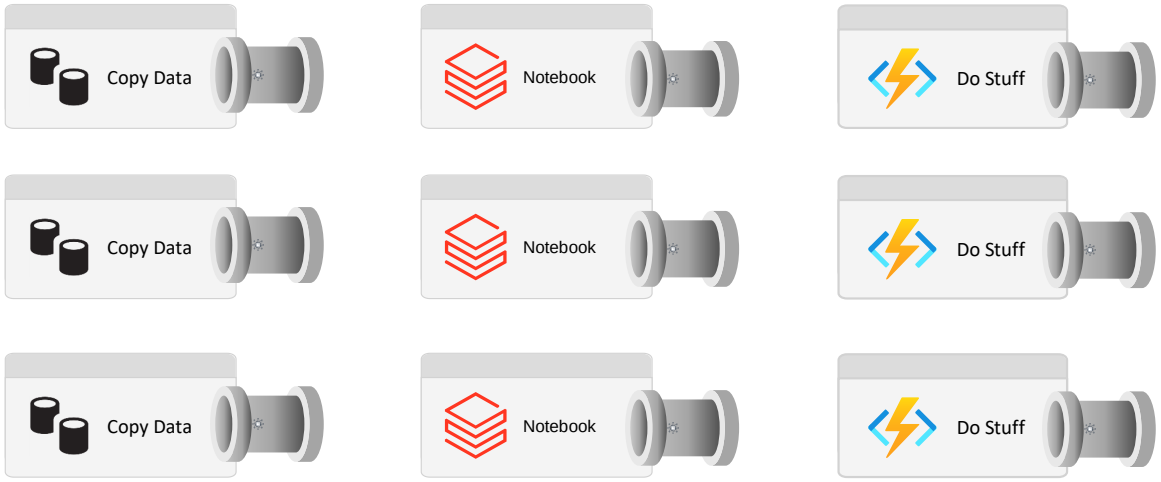
Stage	Pipeline
1	a
1	b
1	c
2	d
2	e
3	f
3	g
3	h
3	i

One More Problem



Calling Our Worker Pipelines



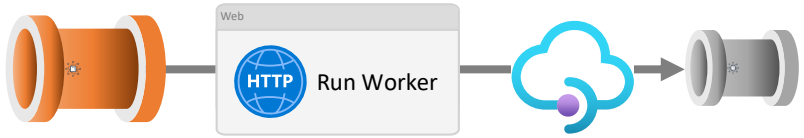


Stages
1
2
3

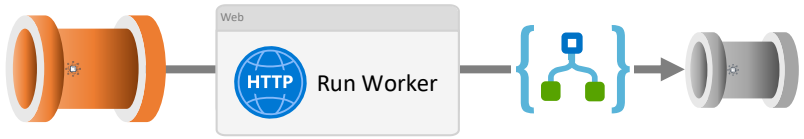
Pipelines
a
b
c
d
e
f
g
h
i

Stage	Pipeline
1	a
1	b
1	c
2	d
2	e
3	f
3	g
3	h
3	i

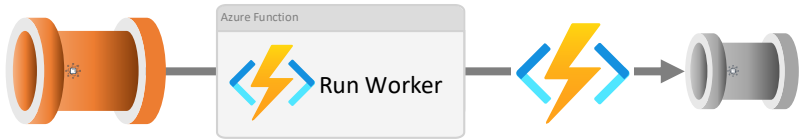
Option 1:



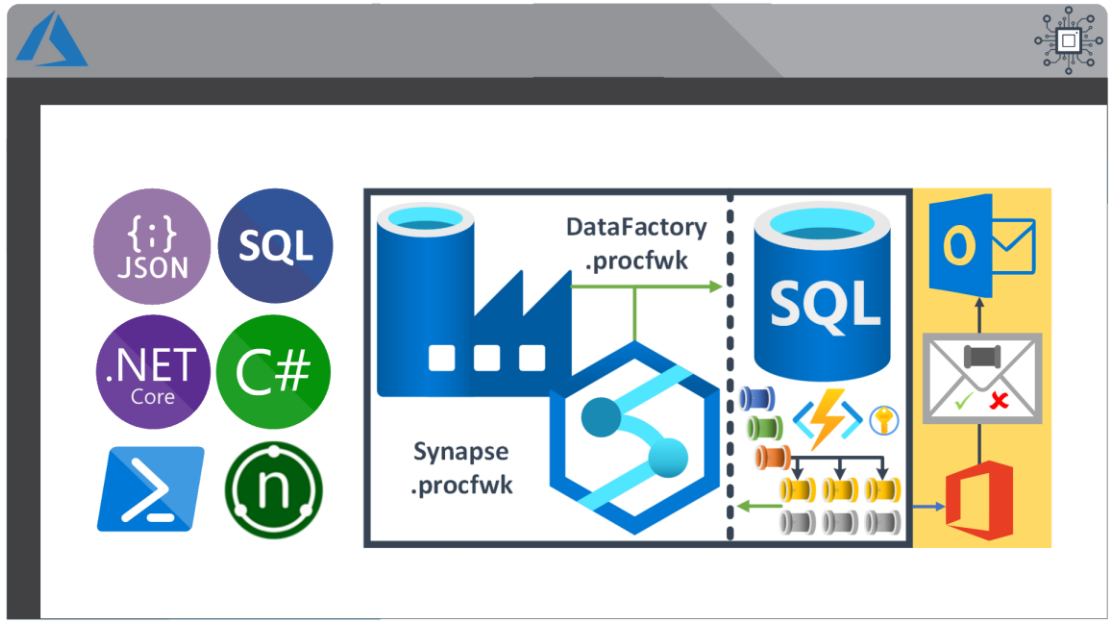
Option 2:



Option 3:

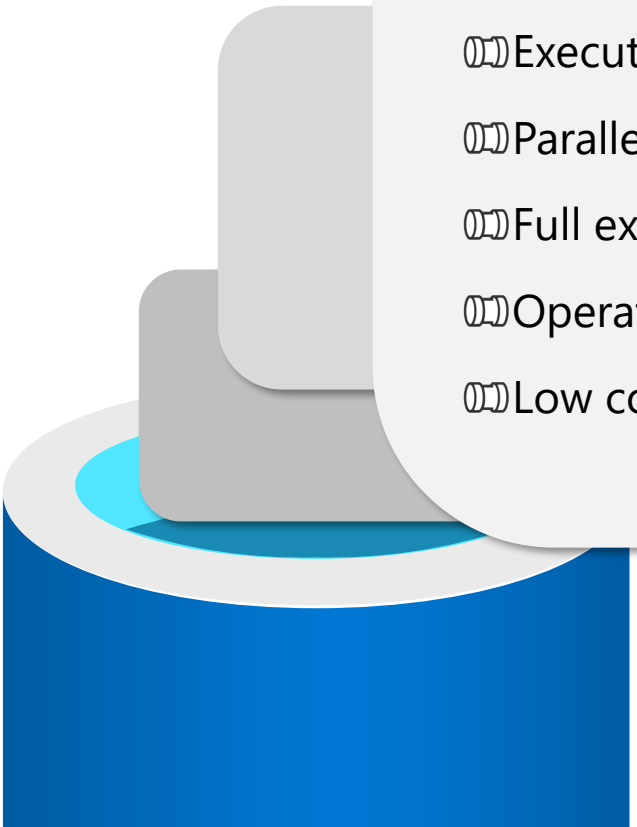


Introducing procfwk.com

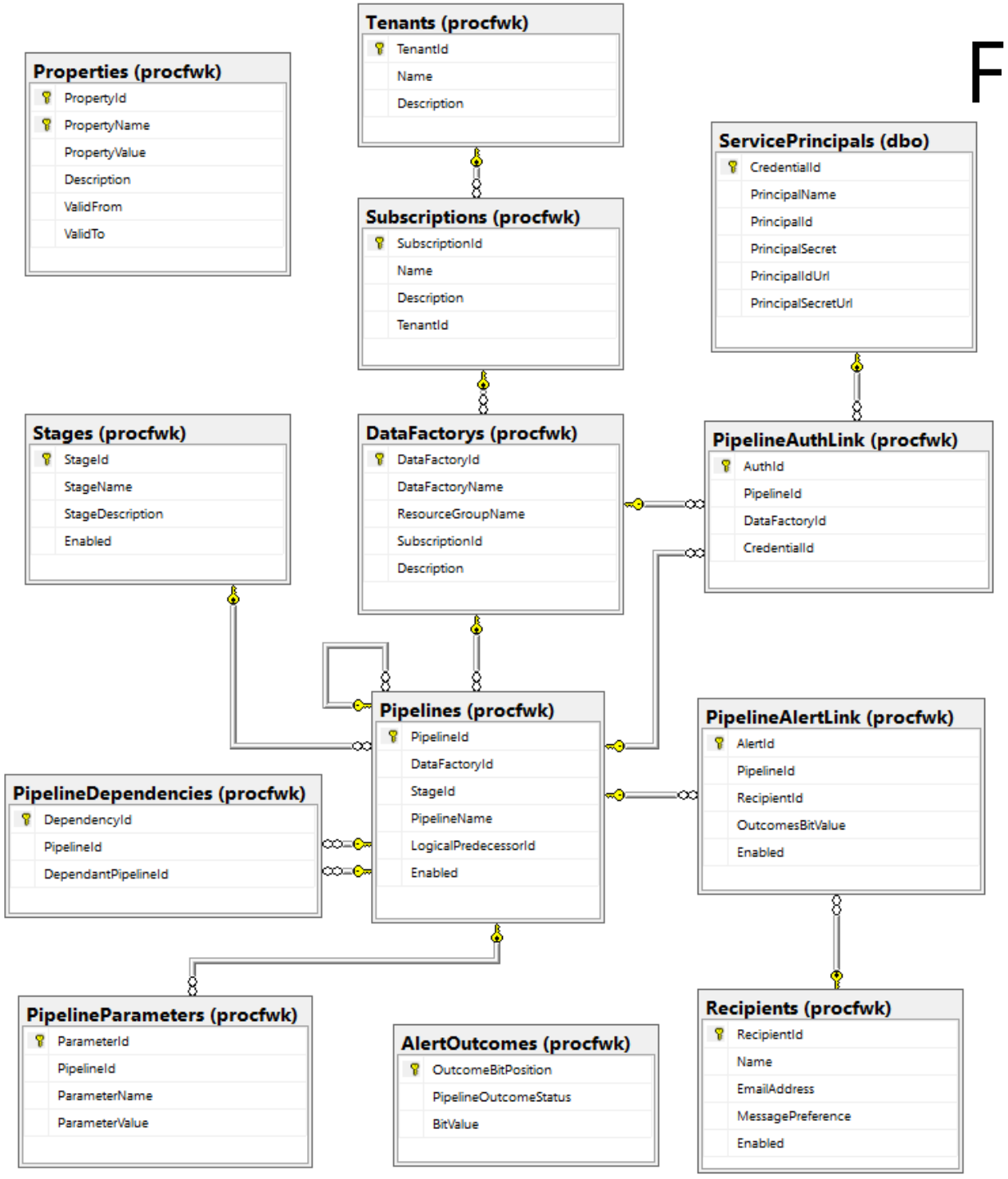


Procfwk Features

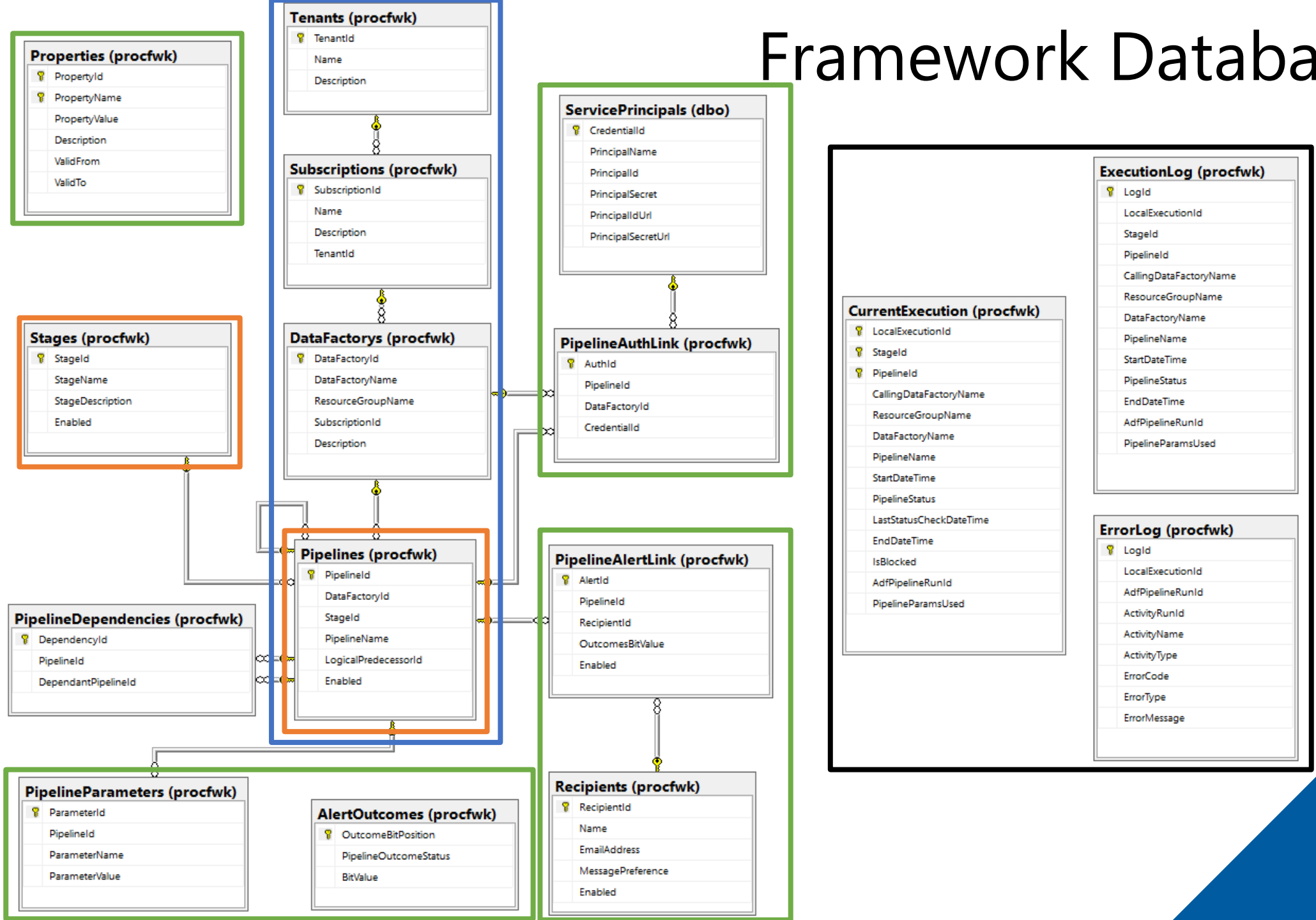
- Granular metadata control.
- Metadata integrity checking.
- Global properties.
- Complete pipeline dependency chains.
- Execution restart-ability.
- Parallel execution.
- Full execution and error logs.
- Operational dashboards.
- Low cost orchestration.
- Disconnection between framework and worker pipelines.
- Cross Tenant/Subscription/Data Factory control flows.
- Pipeline parameter support.
- Simple troubleshooting.
- Easy deployment.
- Email alerting.
- Automated testing.
- Azure Key Vault integration.



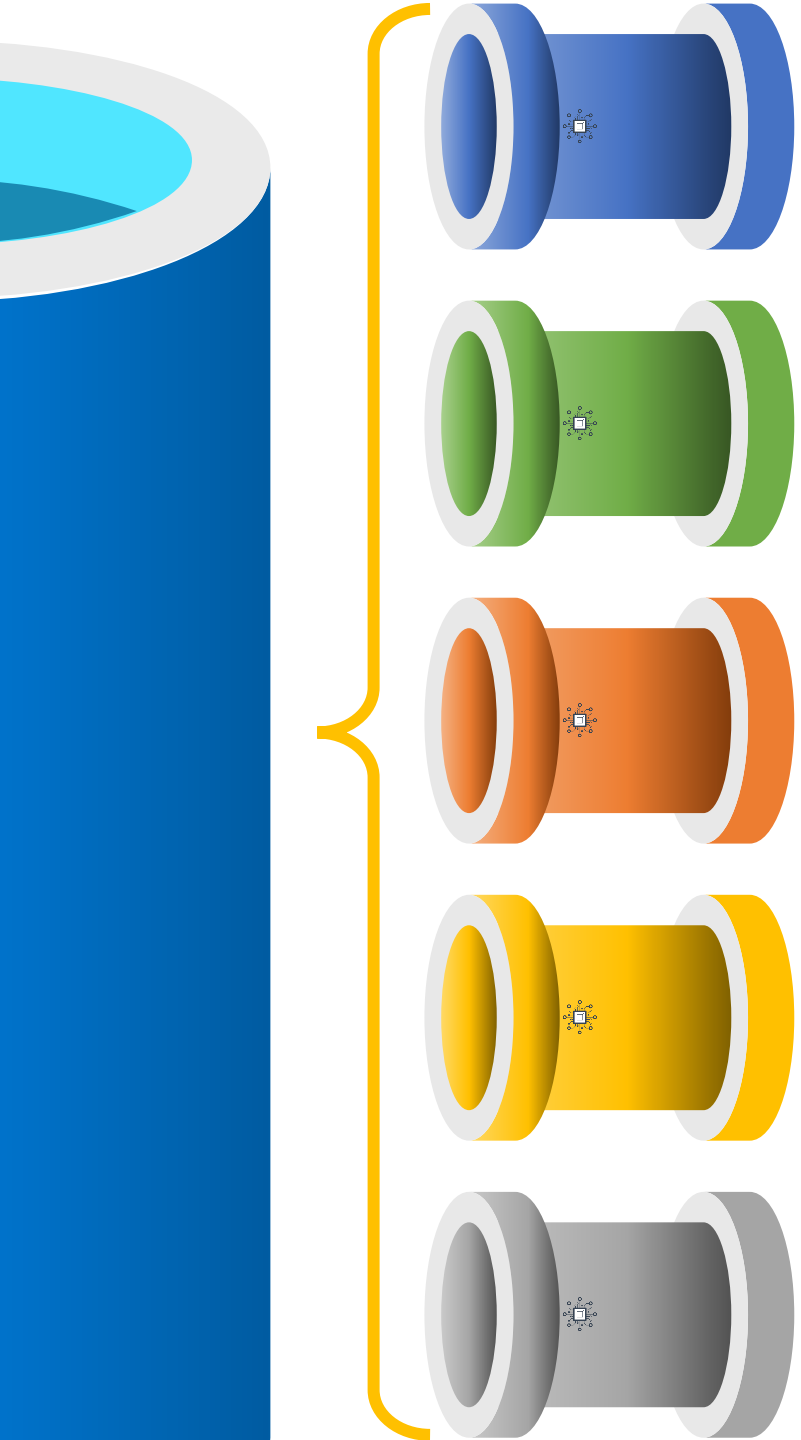
Framework Database



Framework Database



Pipeline Hierarchy



- Grandparent

Role: Optional level platform setup, for example, scale up/out compute services ready for the framework to run.

- Parent

Role: Execution run wrapper and execution stage iterator.

- Child

Role: Scale out triggering of worker pipelines within the execution stage.

- Infant

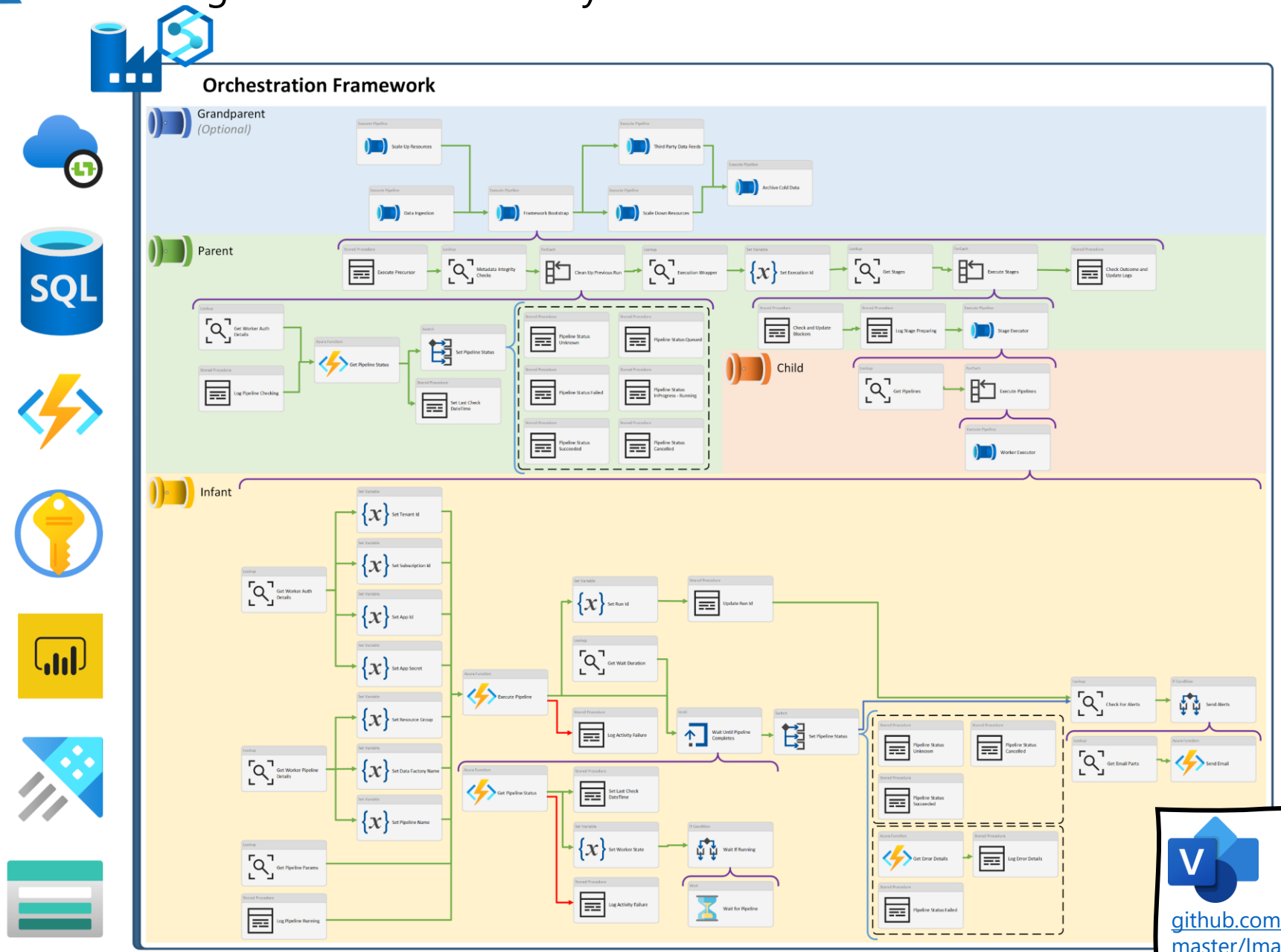
Role: Worker executor, monitor and reporting of the outcome for the single worker pipeline.

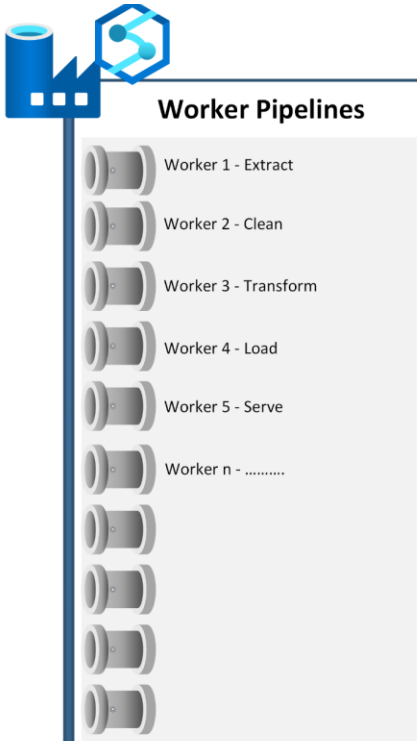
- Worker

Role: Anything specific to the process needing to be performed.

Processing Framework - Activity Chain

Orchestration Framework Support Resources





Go to Visio file in GitHub:
github.com/mrpaulandrew/procfwk/blob/master/Images


DEMO

Module 5:
Metadata Driven Pipelines

- Expressions ✓
- Dynamic Pipeline ✓
- Orchestration Framework ✓