Working in the (Azure Data) Factory the right way

Riccardo Perico

rperico@lucient.com

SQL START! 2022

Riccardo Perico

BI & Power BI Engineer @ Lucient

PBIUG Italy Co-leader

Microsoft Data Platform MVP

Linkedin: https://linkedin.com/in/riccardo-perico

Email: rperico@lucient.com

Slideshare: https://www.slideshare.net/RiccardoPerico

GitHub: https://github.com/R1k91

Partners

UNISS

UNIVERSITÀ
POLITECNICA
DELLE MARCHE

Where are we? We’re in the factory

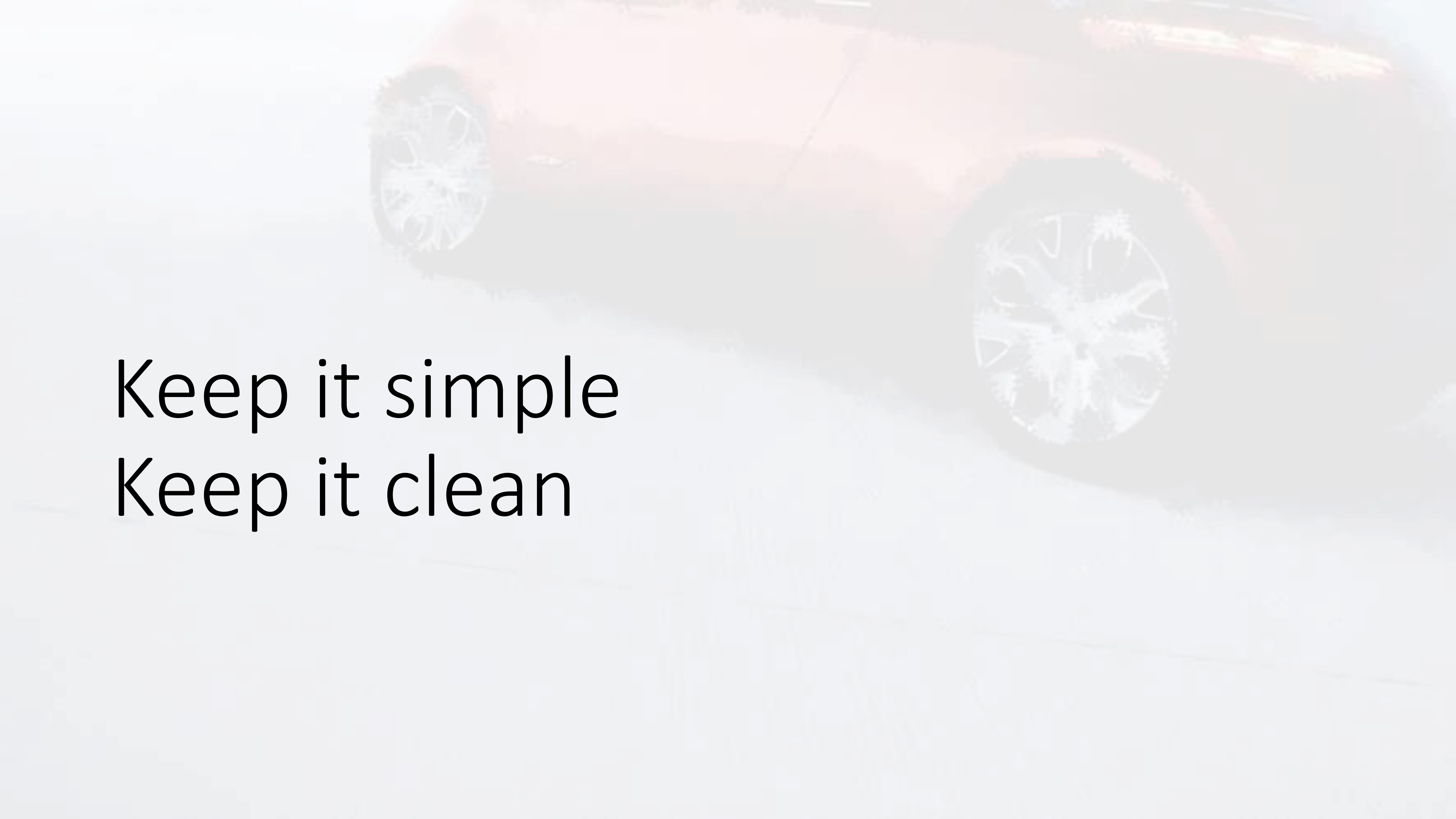
*“**Big data** requires a service that can **orchestrate** and **operationalize** processes to refine these **enormous stores of raw data** into actionable business insights. Azure Data Factory is a **managed cloud service** that's built for these complex hybrid extract-transform-load (**ETL**), extract-load-transform (**ELT**), and **data integration** projects.”*

What about this session?

Starting to work in ADF it's easy but…

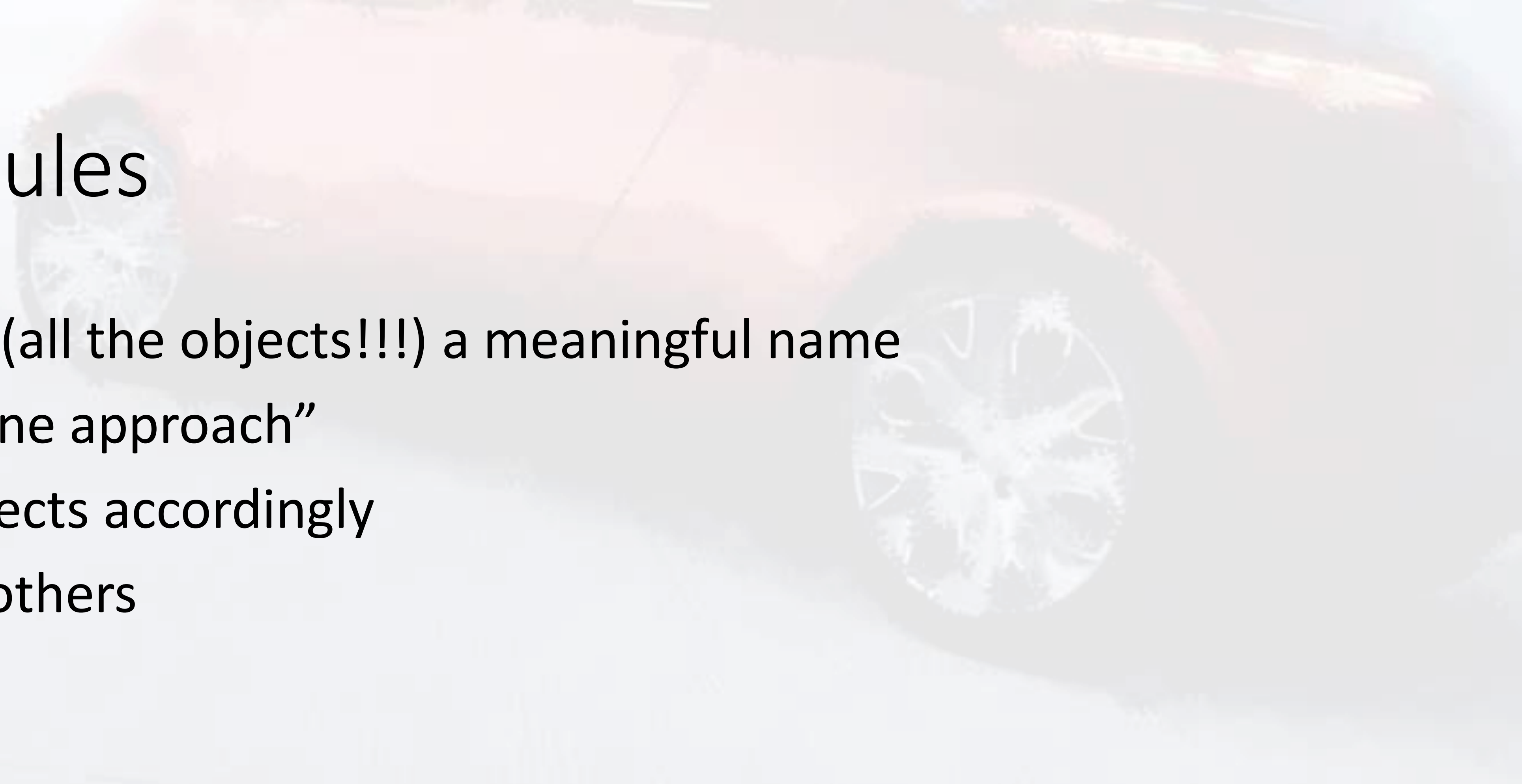
You'll learn by doing especially best practices and patterns

Let me share some lessons learnt from by background



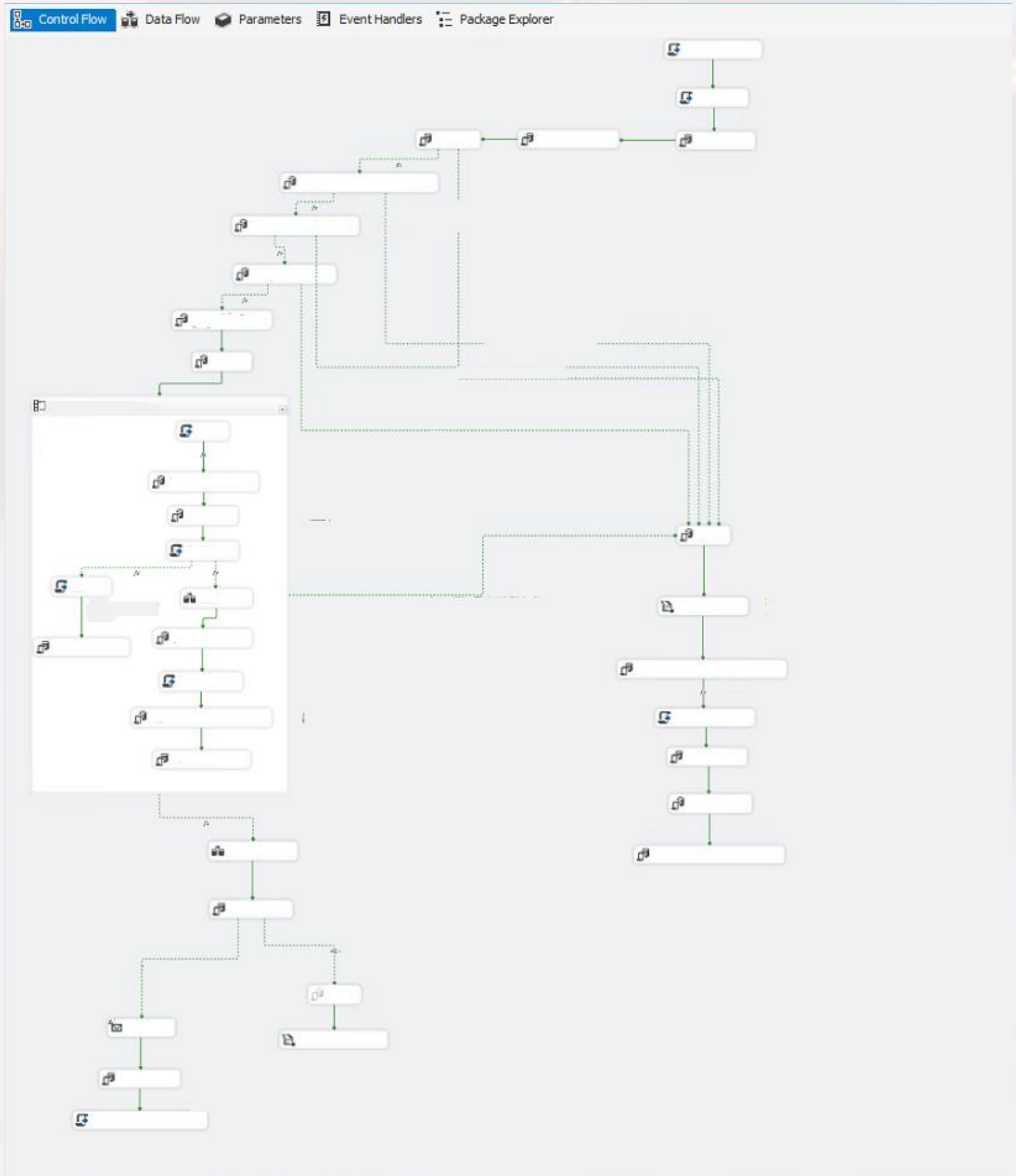
Keep it simple
Keep it clean

General rules

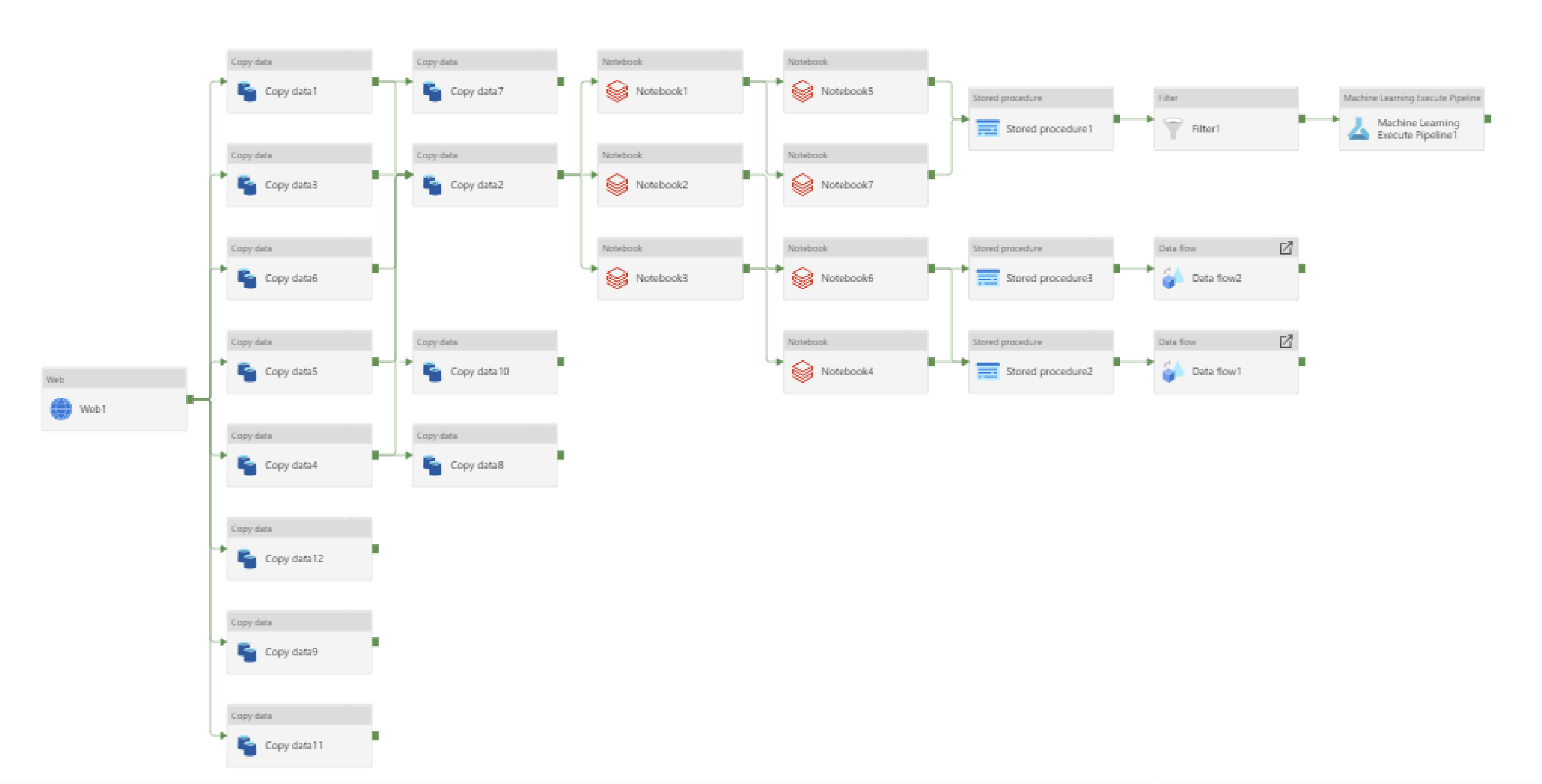


- Give objects (all the objects!!!) a meaningful name
- Use a “pipeline approach”
- Organize objects accordingly
- …and many others

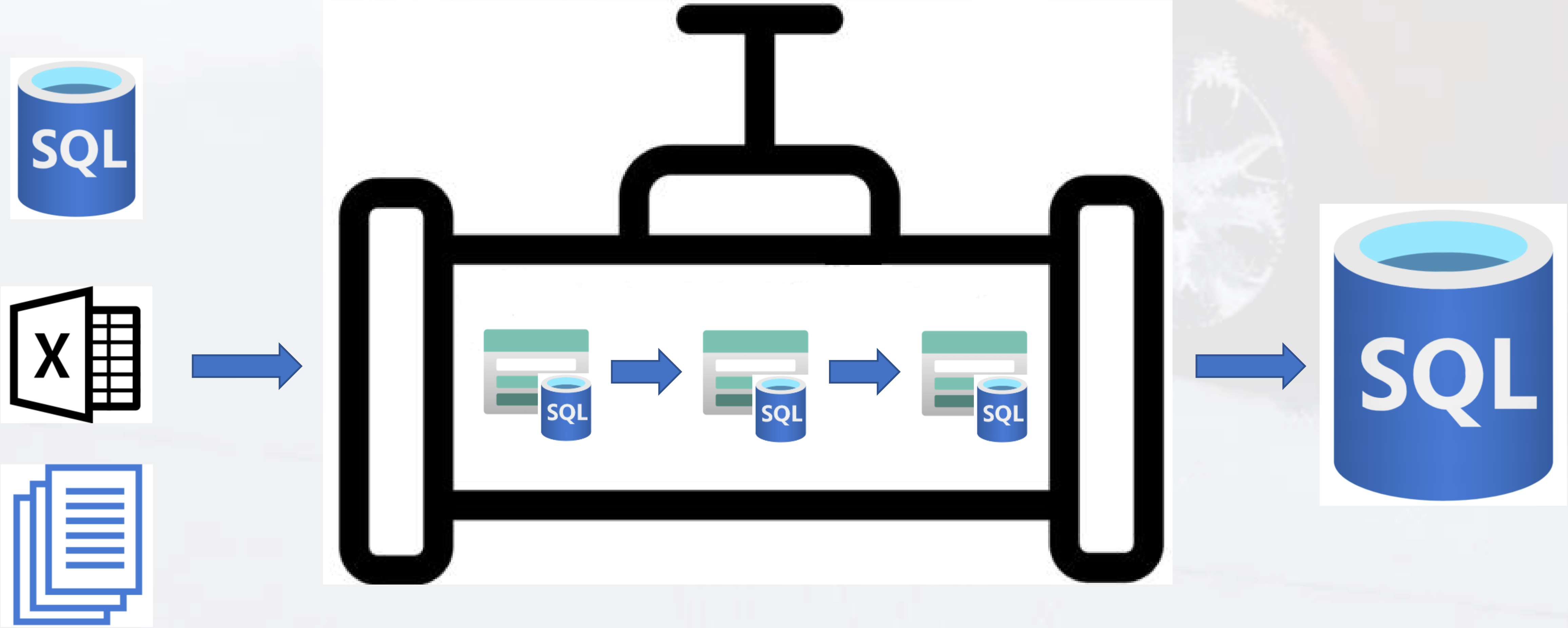
Amarcord


Control Flow
Data Flow
Parameters
Event Handlers
Package Explorer

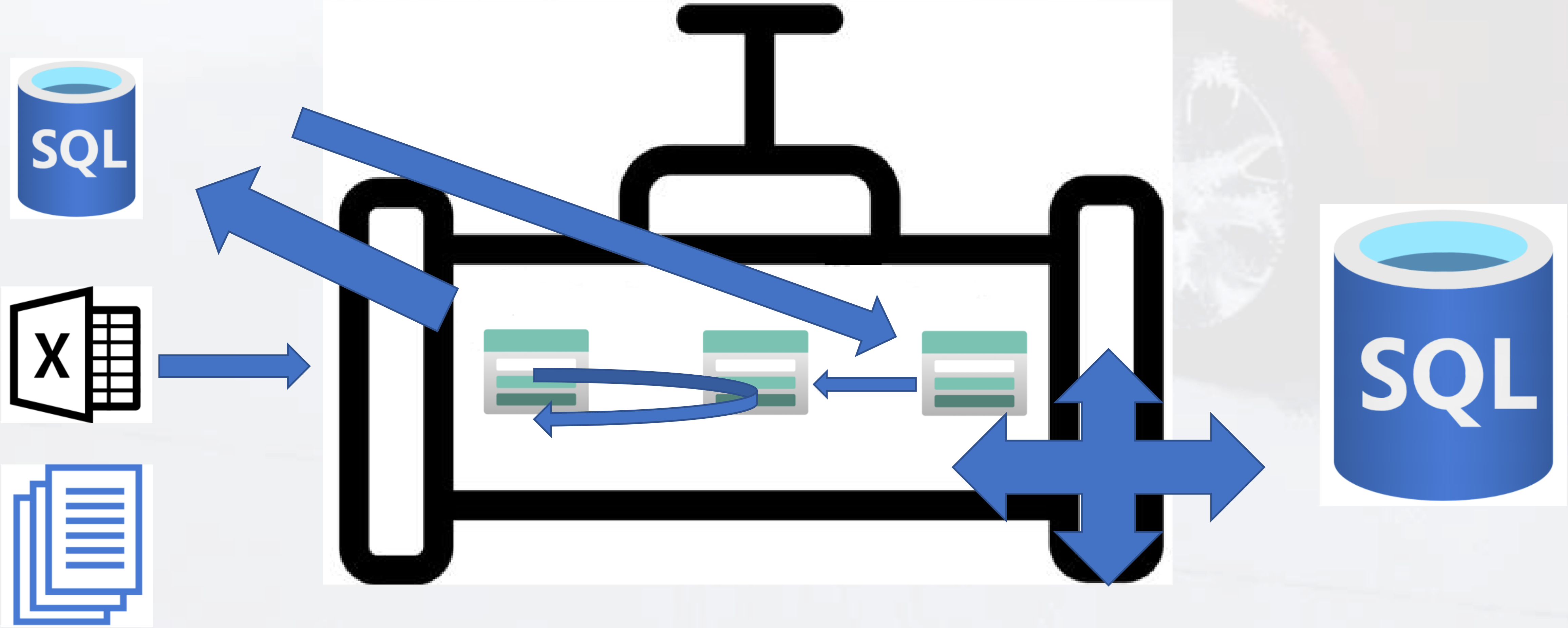
New tools same pitfalls



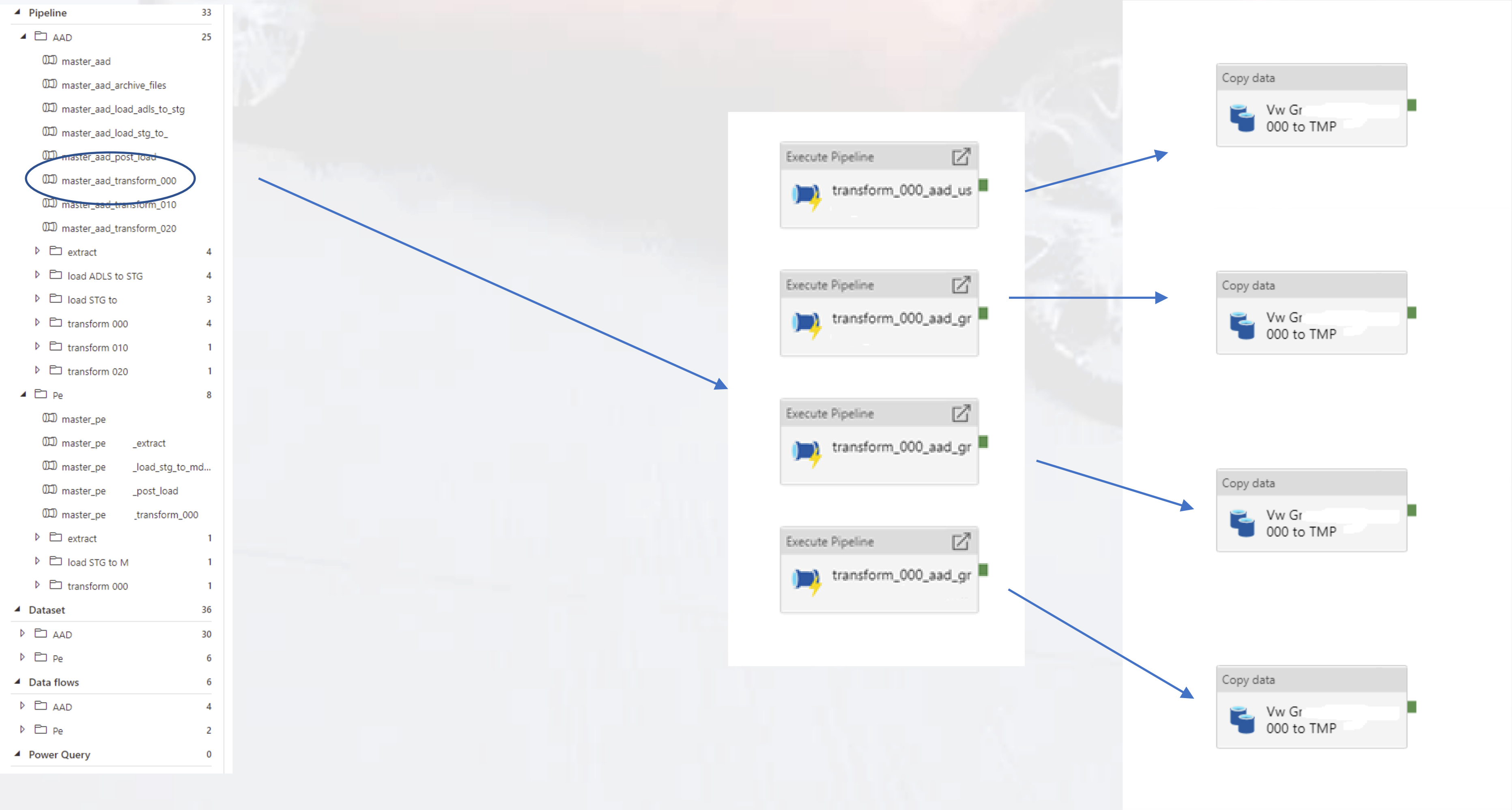
“Pipeline approach”: it flows in one way



Pipeline approach: not that chaos… please



Keep it clean



Pipeline 33
AAD 25
master_aad
master_aad_archive_files
master_aad_load_adls_to_stg
master_aad_load_stg_to_
master_aad_post_load
master_aad_transform_000
master_aad_transform_010
master_aad_transform_020
extract 4
load ADLS to STG 4
load STG to 3
transform 000 4
transform 010 1
transform 020 1
Pe 8
master_pe
master_pe _extract
master_pe _load_stg_to_md...
master_pe _post_load
master_pe _transform_000
extract 1
load STG to M 1
transform 000 1
Dataset 36
AAD 30
Pe 6
Data flows 6
AAD 4
Pe 2
Power Query 0

Execute Pipeline
transform_000_aad_us

Execute Pipeline
transform_000_aad_gr

Execute Pipeline
transform_000_aad_gr

Execute Pipeline
transform_000_aad_gr

Copy data
Vw Gr
000 to TMP

Copy data
Vw Gr
000 to TMP

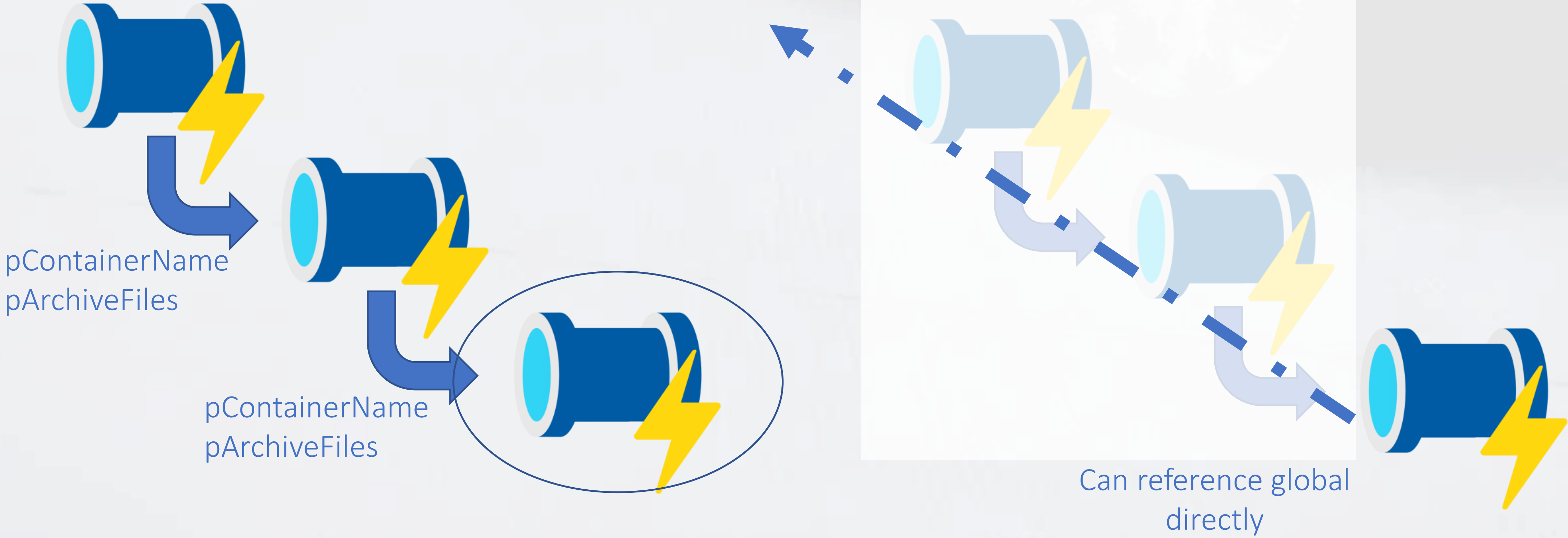
Copy data
Vw Gr
000 to TMP

Copy data
Vw Gr
000 to TMP

Global Parameters

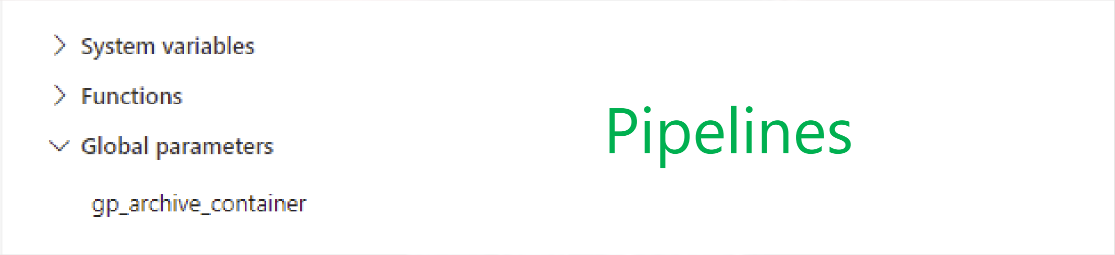
Before and After

Parameter Name	Parameter Value
pContainerName	myContainer
pArchiveFiles	True



One drawback

- Only pipelines can use them



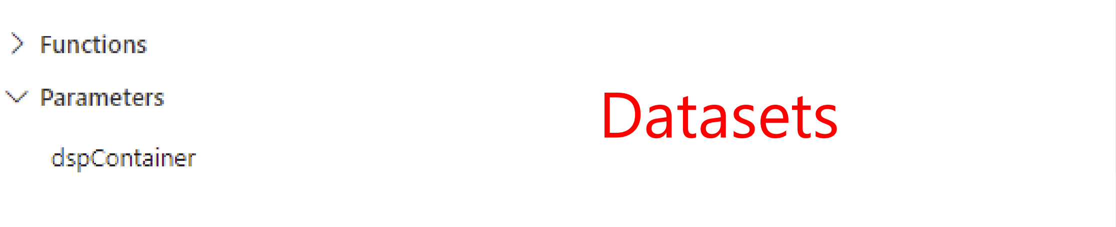
> System variables

> Functions

﹀ Global parameters

gp_archive_container

Pipelines



> Functions

﹀ Parameters

dspContainer

Datasets

Global Parameters

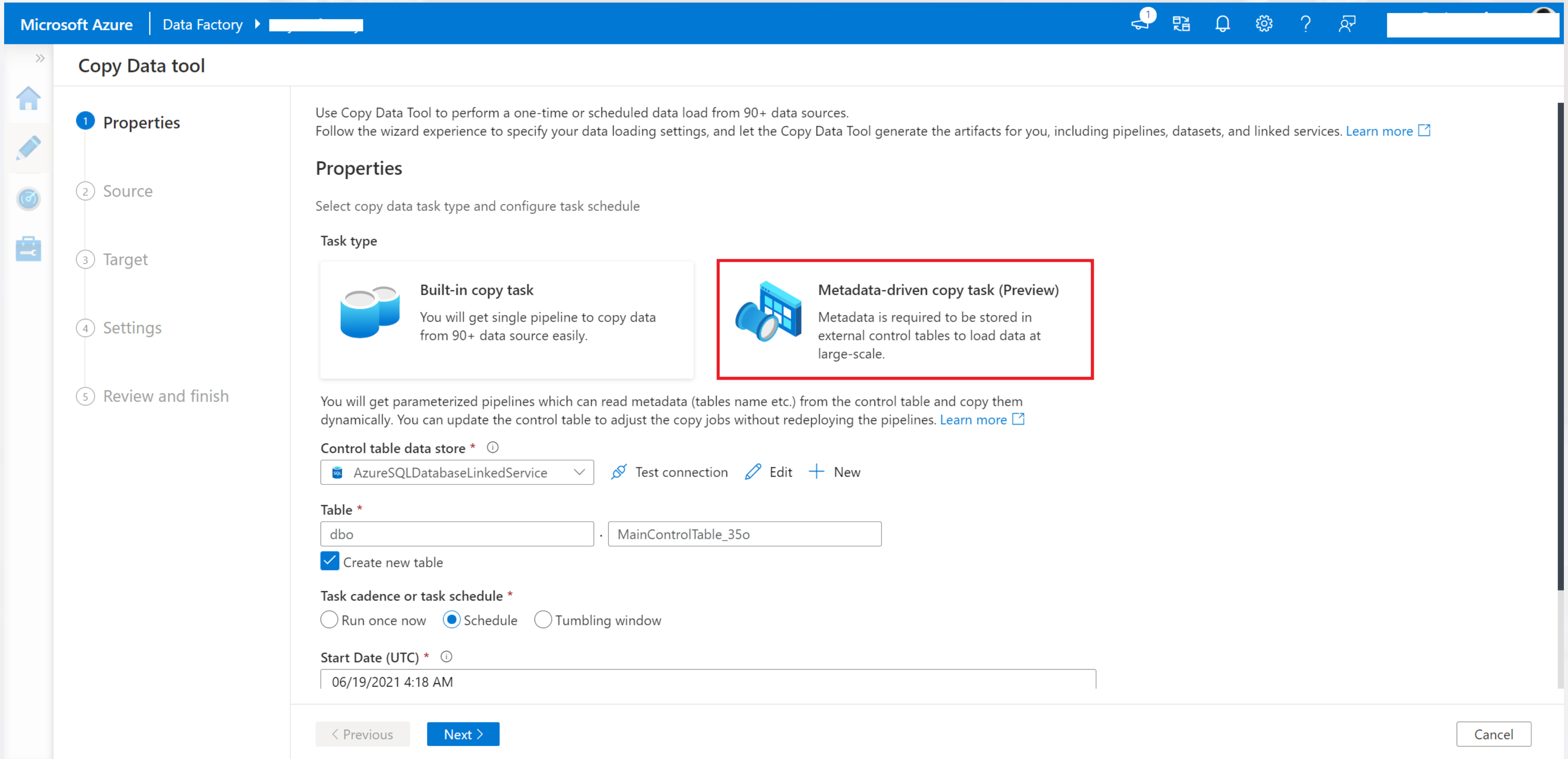
Demo

Metadata Driven Approach

What is a Metadata-driven approach?

- Don’t do it manually
- Make it flexible and dynamic
- Make it configurable

Is there anything out-of-the-box?



Copy Data tool

Use Copy Data Tool to perform a one-time or scheduled data load from 90+ data sources.
Follow the wizard experience to specify your data loading settings, and let the Copy Data Tool generate the artifacts for you, including pipelines, datasets, and linked services. Learn more

Properties

Select copy data task type and configure task schedule

Task type

Built-in copy task
You will get single pipeline to copy data from 90+ data source easily.

Metadata-driven copy task (Preview)
Metadata is required to be stored in external control tables to load data at large-scale.

You will get parameterized pipelines which can read metadata (tables name etc.) from the control table and copy them dynamically. You can update the control table to adjust the copy jobs without redeploying the pipelines. Learn more

Control table data store * : AzureSQLDatabaseLinkedService

Table * : dbo . MainControlTable_35o

☑ Create new table

Task cadence or task schedule *

○ Run once now ● Schedule ○ Tumbling window

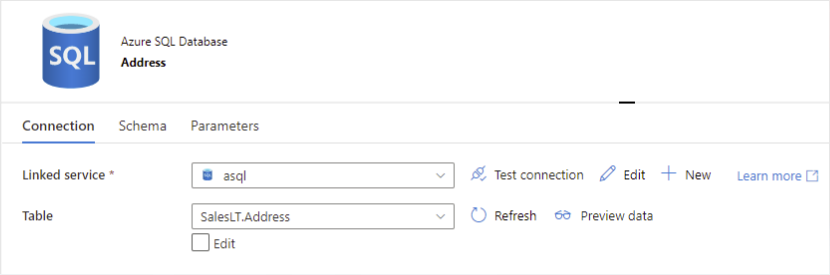
Start Date (UTC) * : 06/19/2021 4:18 AM

Metadata-driven Approach

Demo

Parametrizing Datasets

Parametrizing Datasets

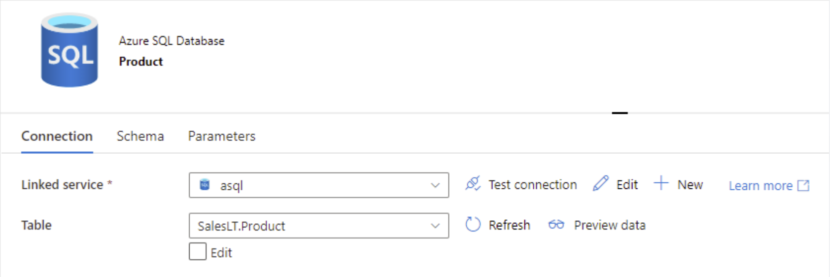

Azure SQL Database
Address

Connection | Schema | Parameters

Linked service * asql | Test connection | Edit | New | Learn more

Table SalesLT.Address | Refresh | Preview data

□ Edit

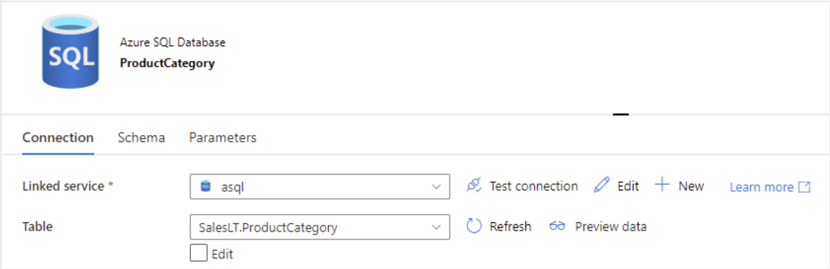

Azure SQL Database
Product

Connection | Schema | Parameters

Linked service * asql | Test connection | Edit | New | Learn more

Table SalesLT.Product | Refresh | Preview data

□ Edit

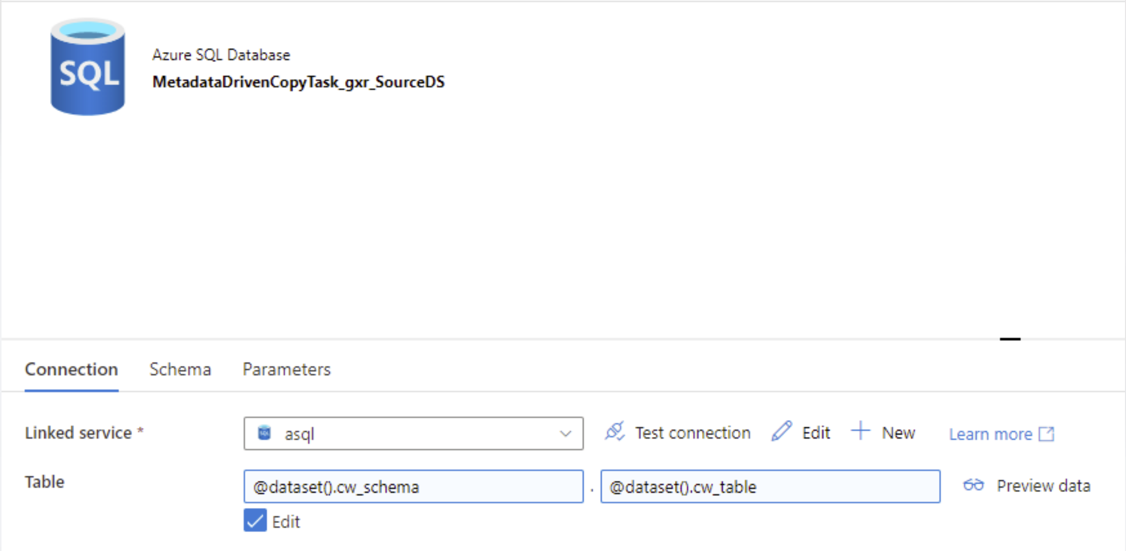

Azure SQL Database
ProductCategory

Connection | Schema | Parameters

Linked service * asql | Test connection | Edit | New | Learn more

Table SalesLT.ProductCategory | Refresh | Preview data

□ Edit


Azure SQL Database
MetadataDrivenCopyTask_gxr_SourceDS

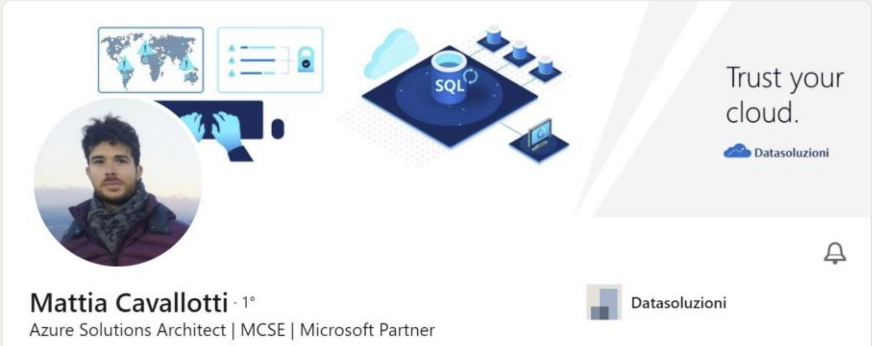
Connection | Schema | Parameters

Linked service * asql | Test connection | Edit | New | Learn more

Table @dataset().cw_schema . @dataset().cw_table | Preview data

☑ Edit

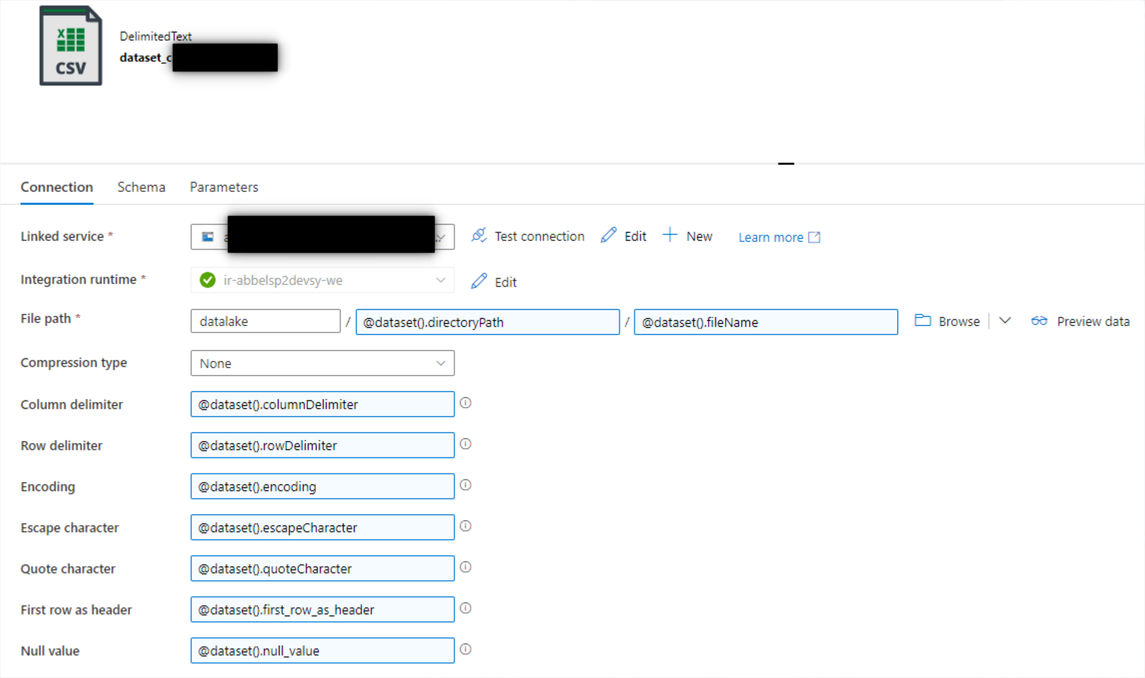
Moreover…



Mattia Cavallotti · 1°

Azure Solutions Architect | MCSE | Microsoft Partner

Datasoluzioni



DelimitedText

dataset_[redacted]

Linked service *	[redacted]
Integration runtime *	ir-abbelsp2devsy-we
File path *	datalake / @dataset().directoryPath / @dataset().fileName
Compression type	None
Column delimiter	@dataset().columnDelimiter
Row delimiter	@dataset().rowDelimiter
Encoding	@dataset().encoding
Escape character	@dataset().escapeCharacter
Quote character	@dataset().quoteCharacter
First row as header	@dataset().first_row_as_header
Null value	@dataset().null_value

Parametrizing Datasets

Demo

ADF loves AKV

ADF loves AKV

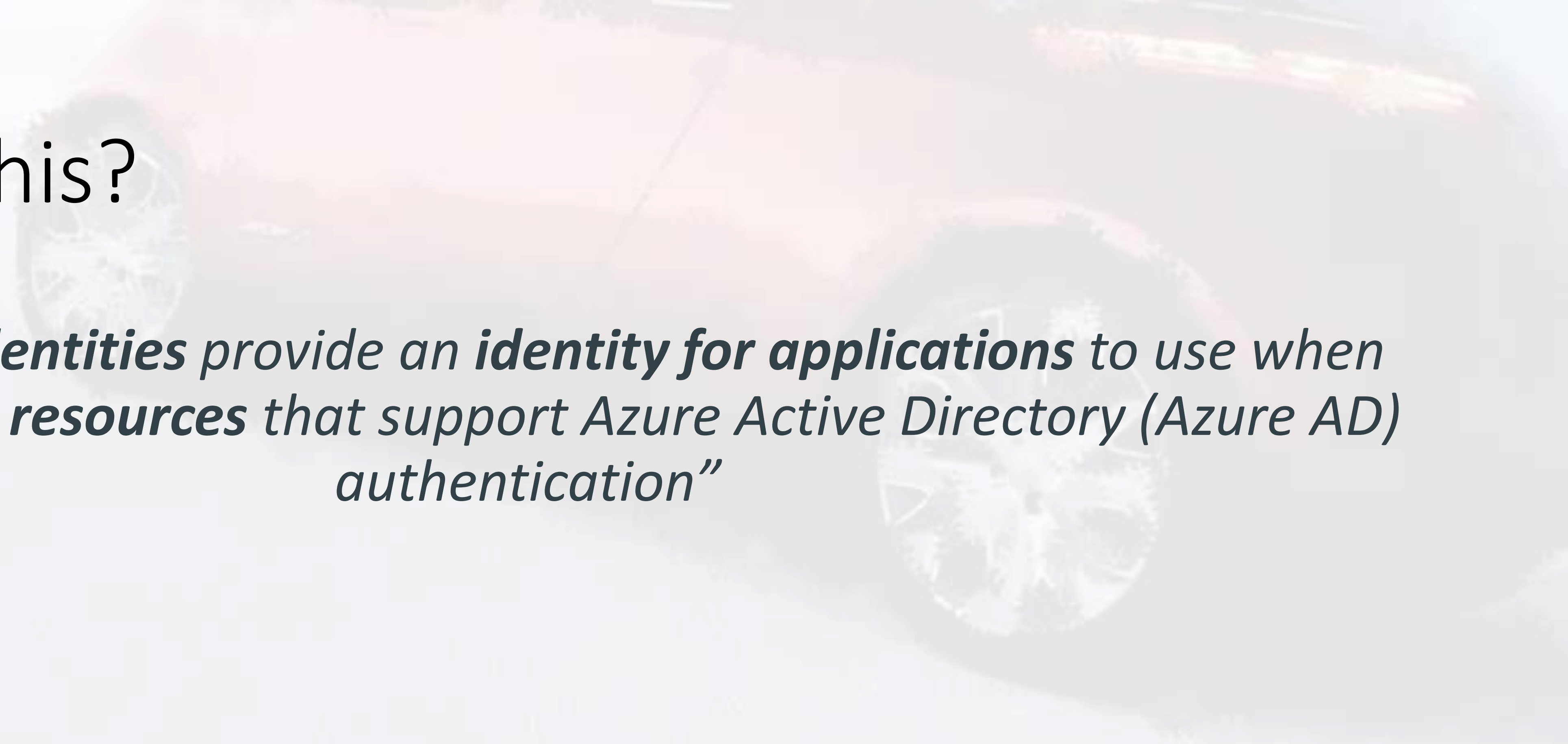
- Best practice: always store **secrets outside ADF**
- Azure Key Vault securely stores and gives access to secrets
- PROS:
  - Developers can work **without knowing secrets**
  - Administrators can **setup and rotate secrets** without accessing ADF
- What should we store?
  - Passwords
  - Users Id / Identities
  - Servers' names / Services’ names
  - …

ADF loves AKV

Demo

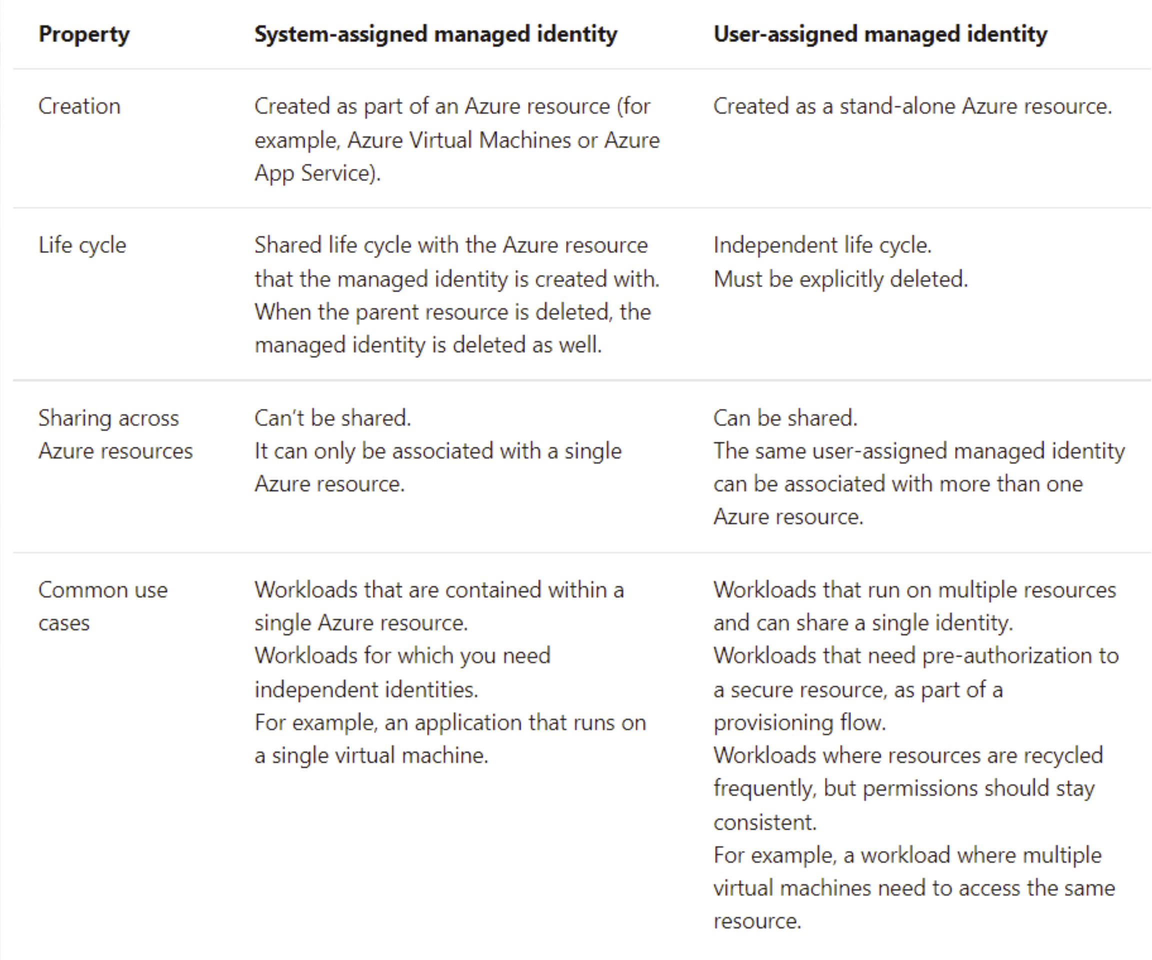
Managed Identities to get access

What is this?



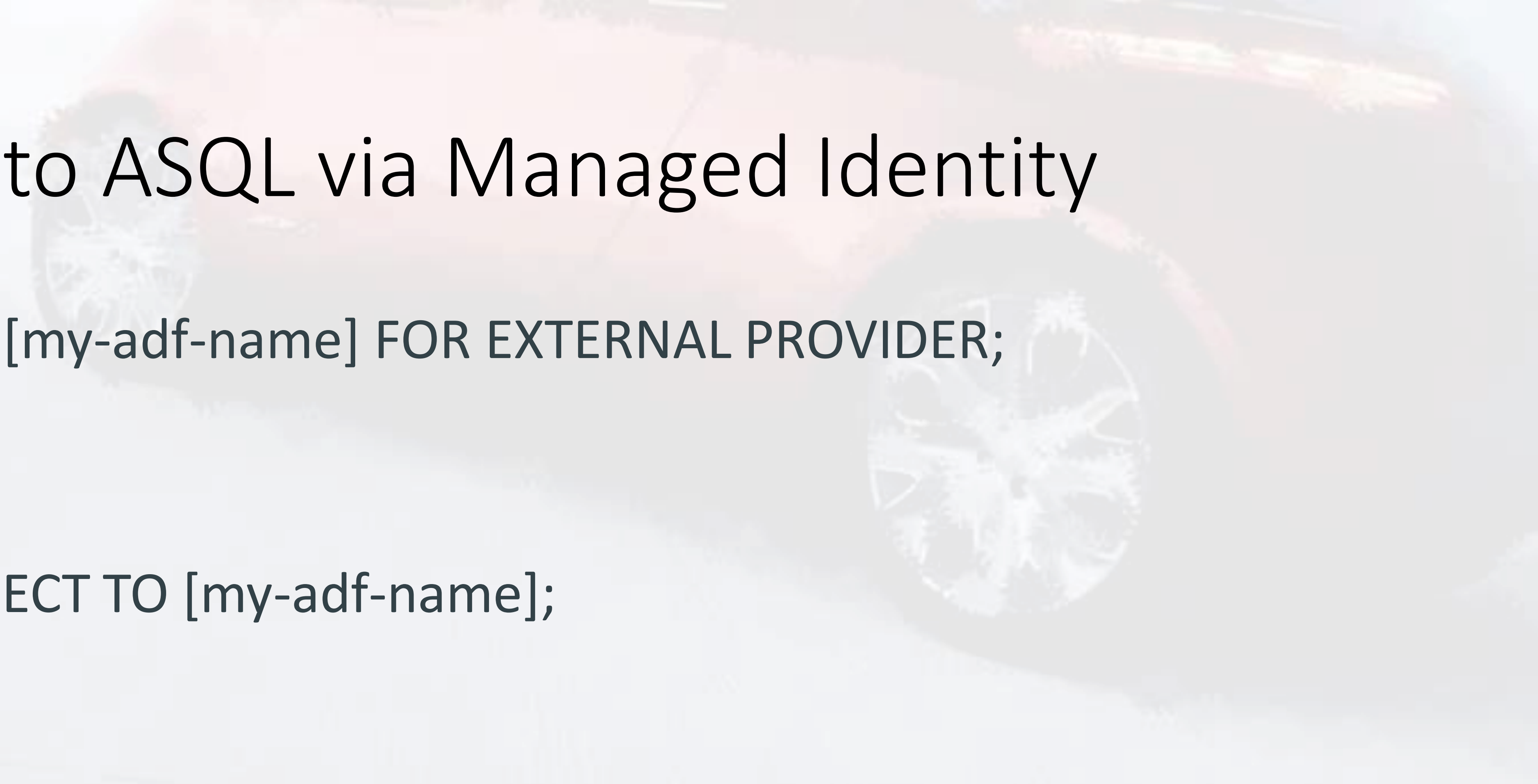
*“**Managed identities** provide an **identity for applications** to use when **connecting to resources** that support Azure Active Directory (Azure AD) authentication”*

2 types and both supported by ADF



Property	System-assigned managed identity	User-assigned managed identity
Creation	Created as part of an Azure resource (for example, Azure Virtual Machines or Azure App Service).	Created as a stand-alone Azure resource.
Life cycle	Shared life cycle with the Azure resource that the managed identity is created with. When the parent resource is deleted, the managed identity is deleted as well.	Independent life cycle. Must be explicitly deleted.
Sharing across Azure resources	Can't be shared. It can only be associated with a single Azure resource.	Can be shared. The same user-assigned managed identity can be associated with more than one Azure resource.
Common use cases	Workloads that are contained within a single Azure resource. Workloads for which you need independent identities. For example, an application that runs on a single virtual machine.	Workloads that run on multiple resources and can share a single identity. Workloads that need pre-authorization to a secure resource, as part of a provisioning flow. Workloads where resources are recycled frequently, but permissions should stay consistent. For example, a workload where multiple virtual machines need to access the same resource.

Connect to ASQL via Managed Identity



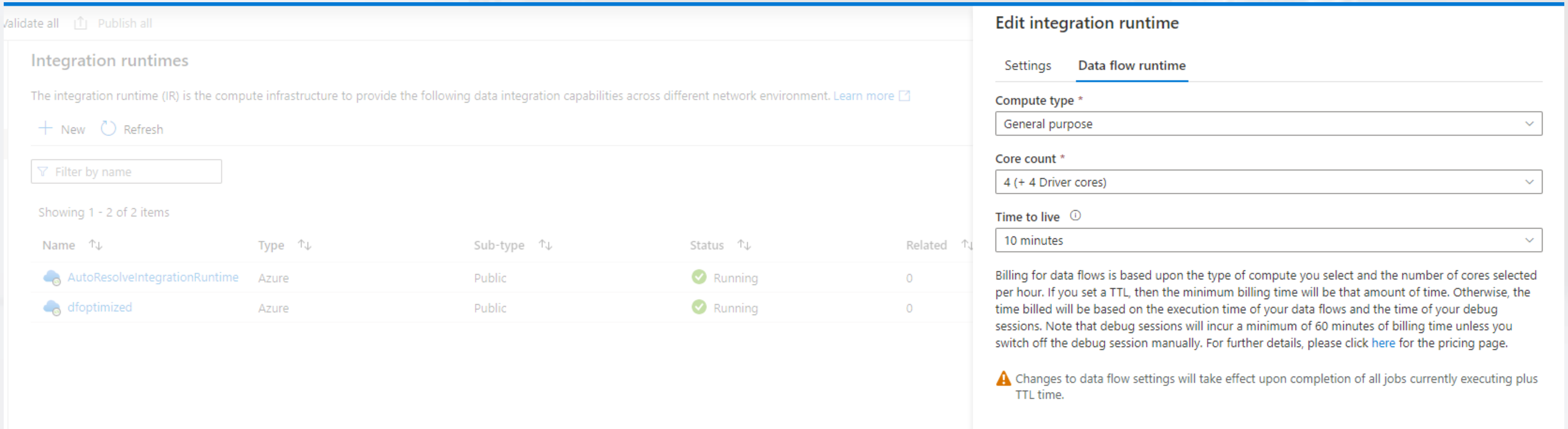
```
CREATE USER [my-adf-name] FOR EXTERNAL PROVIDER;
GO

GRANT CONNECT TO [my-adf-name];
GO

ALTER ROLE [db_owner] ADD MEMBER [my-adf-name];
GO
```

TTL integration runtime

You definitely should use it



Integration runtimes

The integration runtime (IR) is the compute infrastructure to provide the following data integration capabilities across different network environment. Learn more

Showing 1 - 2 of 2 items

Name	Type	Sub-type	Status	Related
AutoResolveIntegrationRuntime	Azure	Public	✓ Running	0
dfoptimized	Azure	Public	✓ Running	0

Edit integration runtime

Compute type *: General purpose

Core count *: 4 (+ 4 Driver cores)

Time to live: 10 minutes

Billing for data flows is based upon the type of compute you select and the number of cores selected per hour. If you set a TTL, then the minimum billing time will be that amount of time. Otherwise, the time billed will be based on the execution time of your data flows and the time of your debug sessions. Note that debug sessions will incur a minimum of 60 minutes of billing time unless you switch off the debug session manually. For further details, please click here for the pricing page.

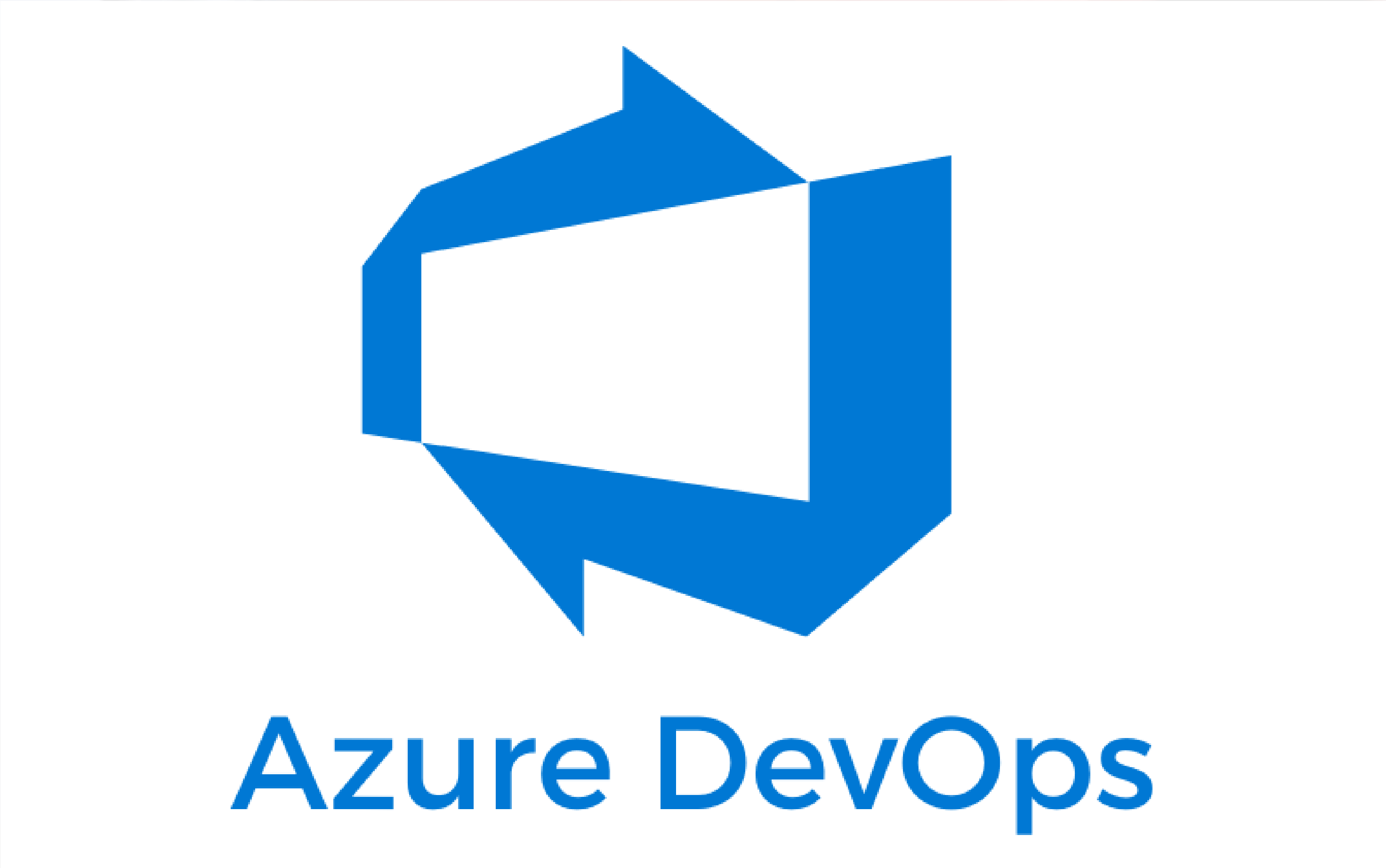
⚠ Changes to data flow settings will take effect upon completion of all jobs currently executing plus TTL time.

TTL integration runtime

Demo

My last advice for today…

No demo sorry…



Useful Links

- ADF docs: http://tiny.cc/adfrw1
- Metadata driven out-of-the-box: http://tiny.cc/adfrw2
- Metadata driven by Paul Andrew: https://github.com/mrpaulandrew/procfwk
- Managed identities: http://tiny.cc/adfrw3
- Connect to ASQL using ADF’s Managed Identity: http://tiny.cc/adfrw4

Q&A

Thank you!

SQL START! 2022