

Les activités

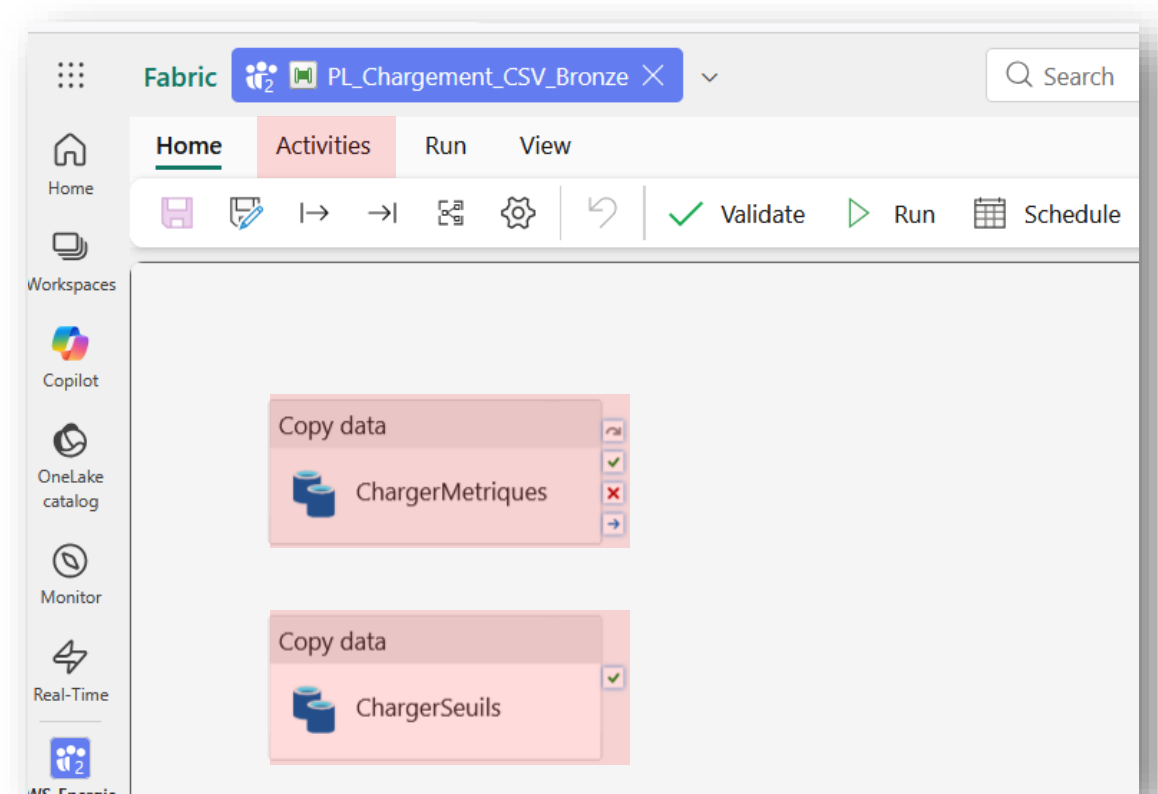
Activités du pipeline Microsoft Fabric Data

Voici la liste des principales **activités** disponibles dans un **pipeline Microsoft Fabric Data Factory**:

Elles sont généralement regroupées en trois grandes catégories :

1. Activités de mouvement de données (Data Movement)

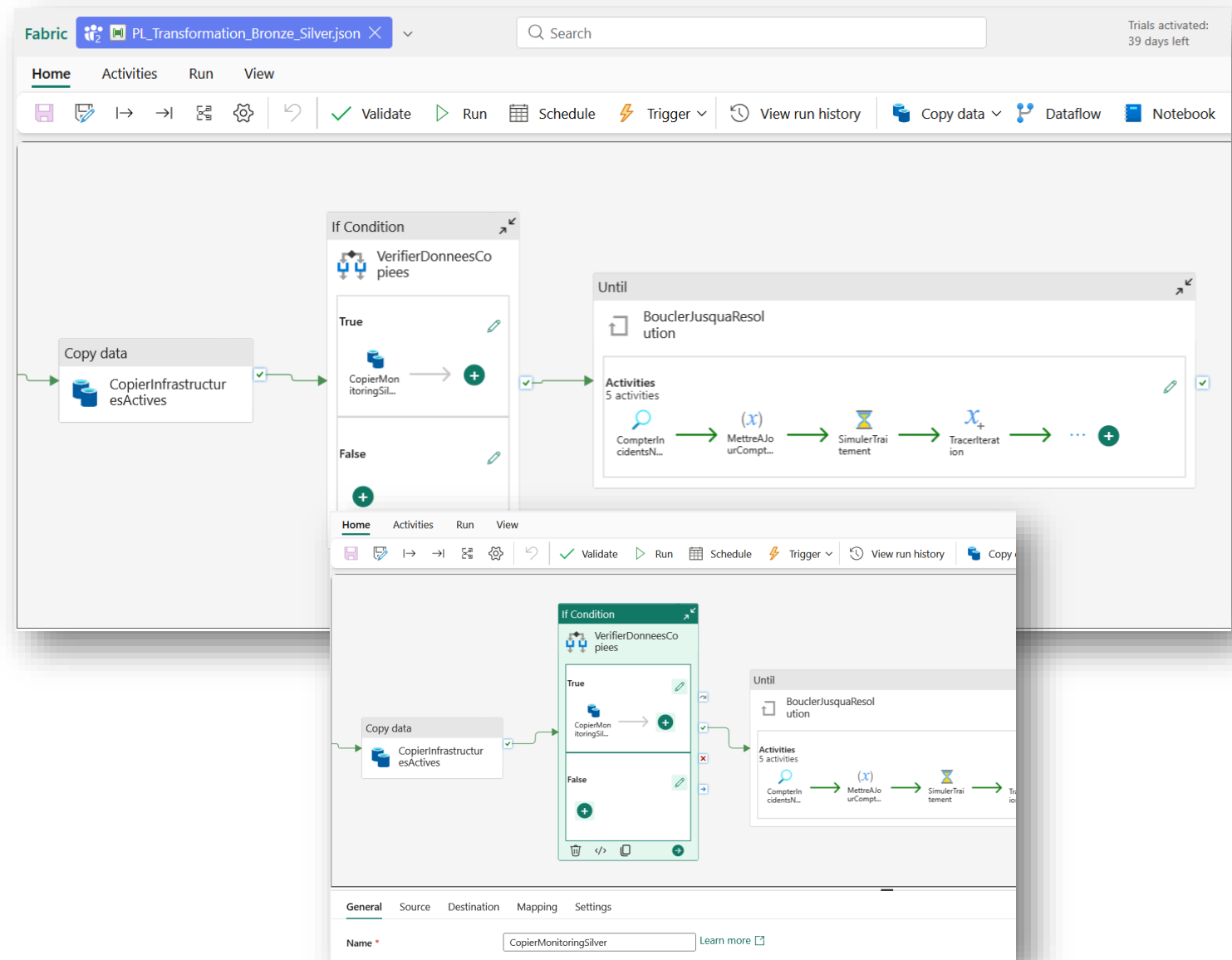
- **Copy** → Copie de données (la plus utilisée – supporte + de 200 connecteurs)
- **Copy Job** (depuis fin 2025, GA) → Version optimisée / simplifiée de Copy pour certains scénarios



Activités du pipeline Microsoft Fabric Data

2. Activités de transformation de données (Data Transformation)

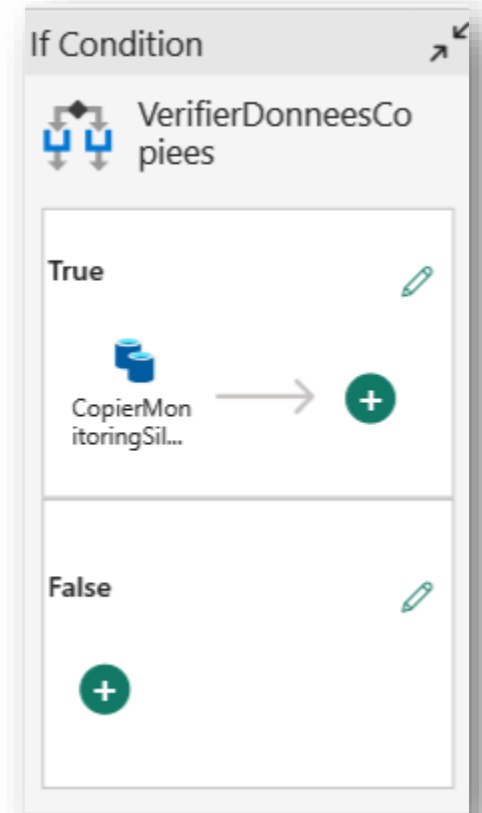
- **Dataflow Gen2** → Exécute un Dataflow Gen2 (transformation low-code)
- **Notebook / Fabric Notebook** → Lance un notebook Spark (PySpark, Spark SQL, Scala...)
- **Spark Job Definition** → Exécute une définition de job Spark pré-enregistrée
- **Data Wrangler** (via pipeline) → Parfois intégré indirectement via Dataflow ou Notebook



Activités du pipeline Microsoft Fabric Data

3. Activités de contrôle de flux (Control Flow)

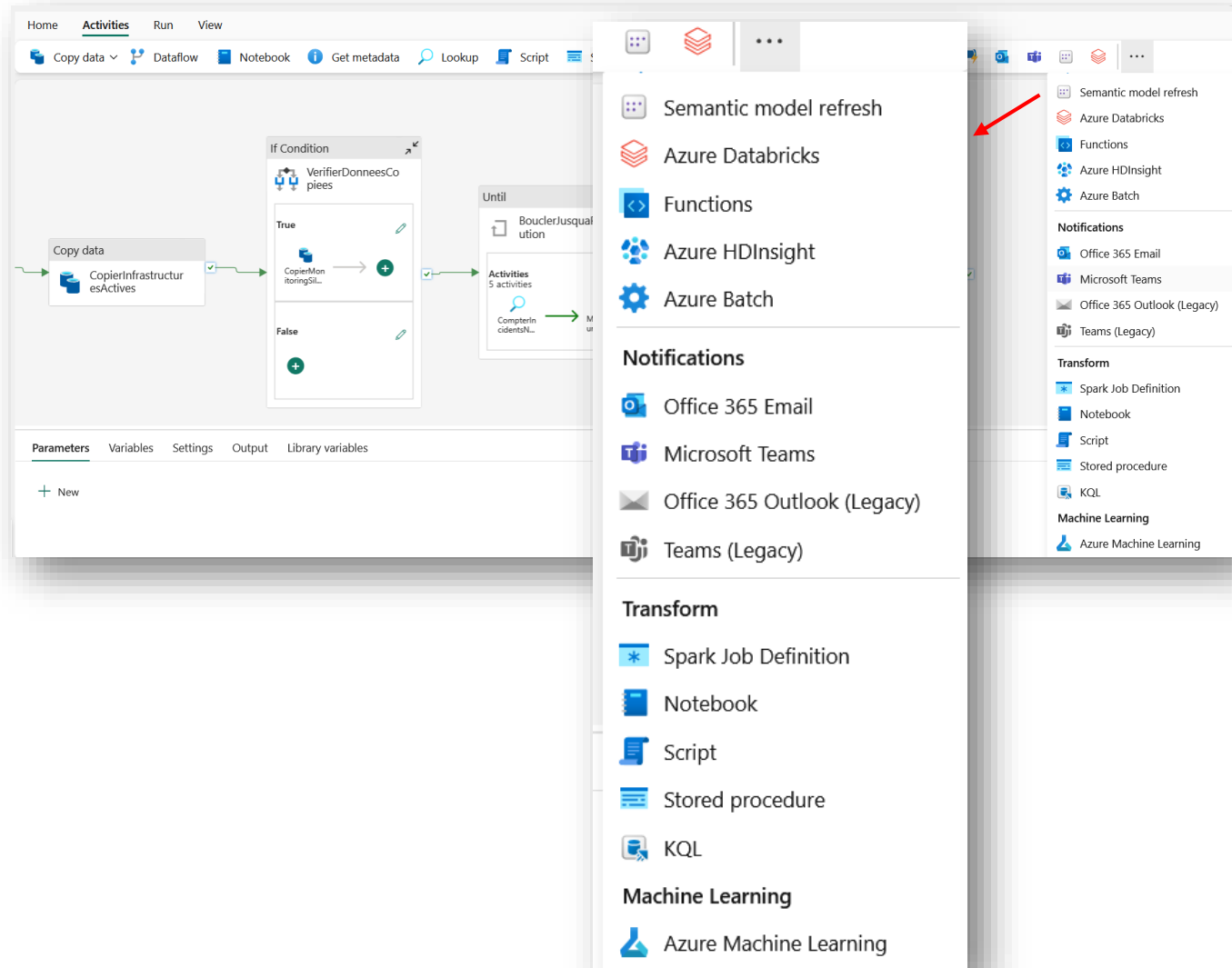
- **ForEach** → Boucle sur une collection (tableaux, listes, résultats de Lookup...)
- **If Condition** → Branchement conditionnel (if/else)
- **Until** → Boucle tant qu'une condition n'est pas remplie
- **Wait** → Pause pendant X secondes/minutes
- **Set Variable** → Modifier la valeur d'une variable
- **Append Variable** → Ajouter une valeur à une variable de type Array
- **Lookup** → Récupérer un résultat (1 ligne ou petite liste) depuis une source
- **Get Metadata** → Récupérer les métadonnées d'un fichier/dossier (existence, taille, last modified...)
- **Web / Web Hook** → Appel HTTP / REST API
- **Invoke Pipeline** → Appeler un autre pipeline (sous-pipeline)
- **Execute Pipeline** (ancien nom parfois vu) → Similaire à Invoke Pipeline



Activités du pipeline Microsoft Fabric Data

Activités les plus couramment utilisées en pratique (top 10)

- Copy
- ForEach
- Lookup
- Dataflow Gen2
- Notebook
- If Condition
- Set Variable
- Get Metadata
- Wait
- Semantic Model Refresh



Paramètres Vs Variables

Variables vs paramètres

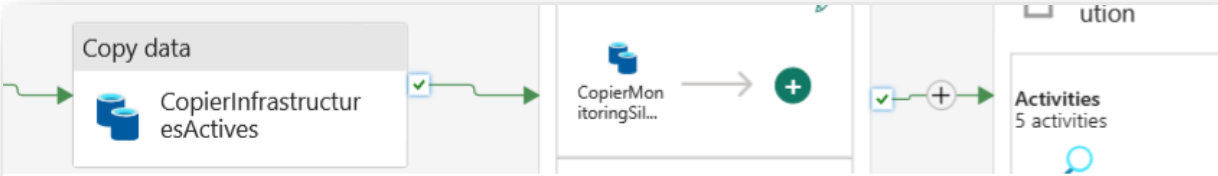
Quand utiliser chacun ?

- **Variables** : Pour la logique de flux (boucles, conditions, calculs dynamiques).
- **Paramètres** : Pour personnaliser l'exécution sans modifier le pipeline (ex : cible de déploiement, chemins de données).

Voici un aperçu des **paramètres** les plus importants pour les activités les plus couramment utilisées dans les **pipelines Microsoft Fabric Data Factory**

Presque toutes les activités partagent des **paramètres généraux** :

- **Name** : nom unique de l'activité
- **Description** : texte optionnel
- **Timeout** : durée max)
- **Retry** : nombre de tentatives
- **Retry interval (sec)** : délai entre retries
- **Secure Input / Secure Output** : masquer les entrées/sorties dans les logs (sécurité)



The screenshot shows a pipeline in Microsoft Fabric Data Factory. The pipeline consists of three activities: 'Copy data', 'CopierInfrastructureActives', and 'Activities 5 activités'. The 'Variables' tab is selected, displaying a table of parameters for the selected activity.

<input type="checkbox"/>	Name	Type	Default value
<input type="checkbox"/>	AnomaliesDetectees	Array	Value
<input type="checkbox"/>	IncidentsNonResolus	Integer	50
<input type="checkbox"/>	NombreIterations	Integer	0

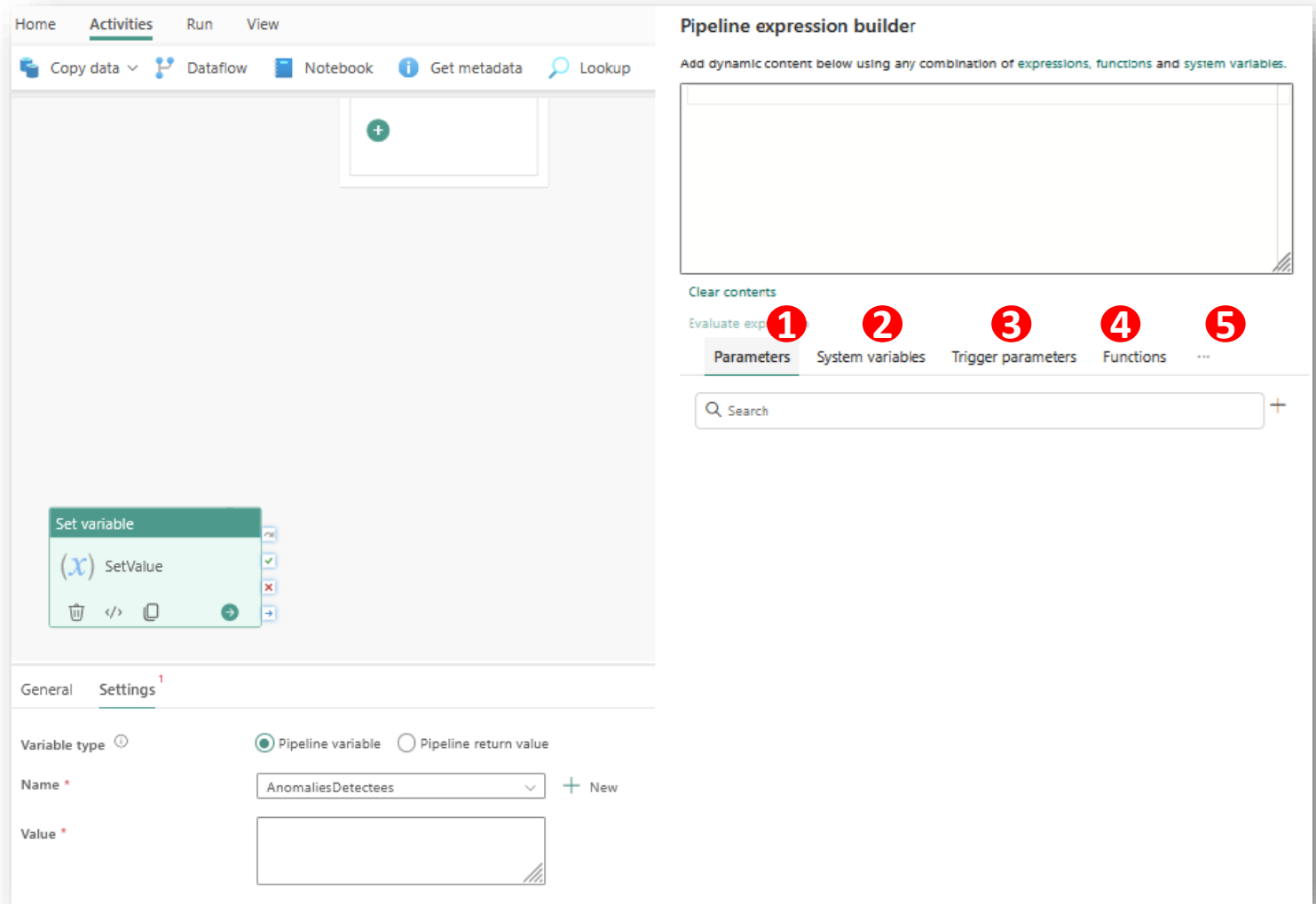
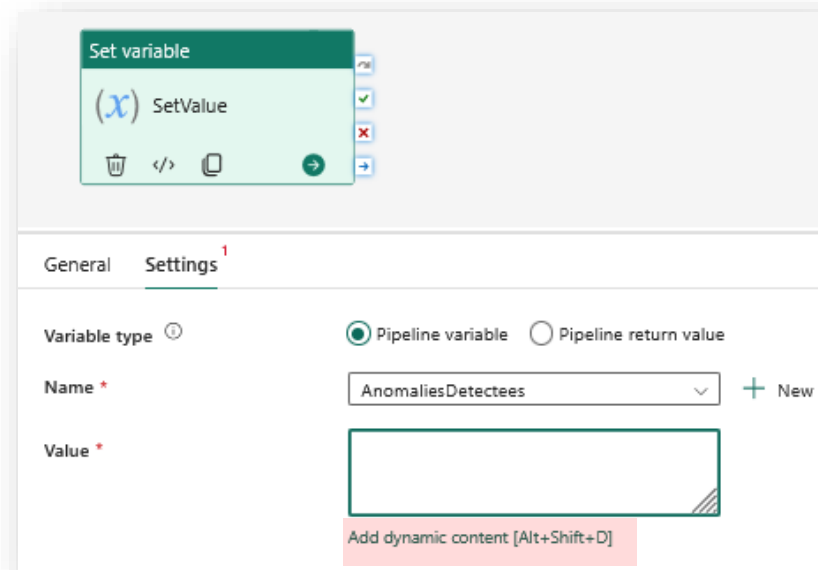
Les expressions

Expressions

Quand utiliser ?

Dans Microsoft Fabric, les expressions dans les pipelines sont utilisées pour créer des valeurs dynamiques.

Les expressions sont définies à travers des activités **Set Variable**



Expressions

Les paramètres ❶

Valeurs définies dans le pipeline
→ Rendent le pipeline réutilisable
et configurable

Ex : `@pipeline().parameters.MonParam`

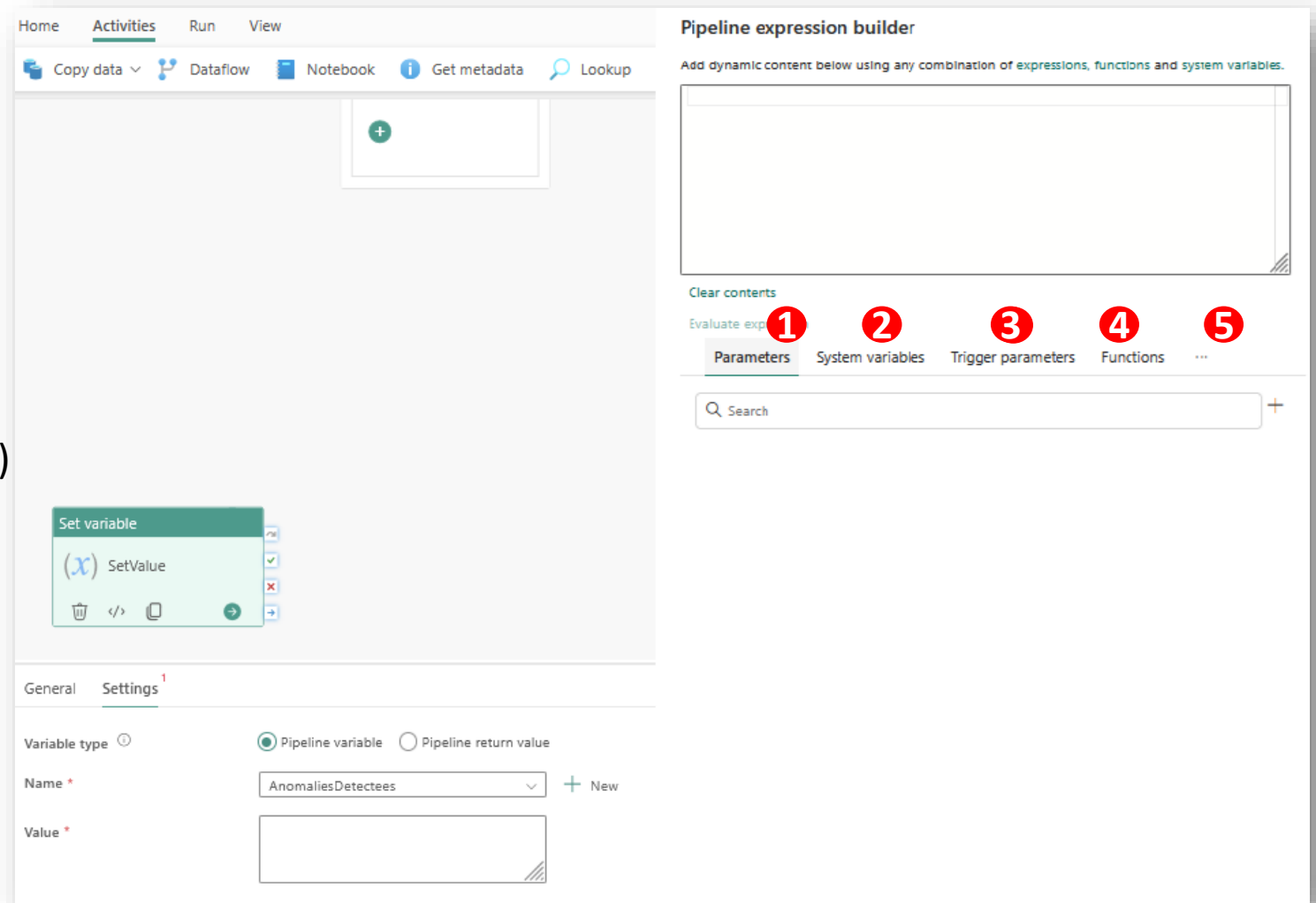
Les System variables ❷

Infos runtime fournies par Fabric (lecture seule)
Exemples :

- `@pipeline().RunId`
- `@pipeline().TriggerTime`
- `@pipeline().TriggerName`

Trigger parameters ❸

Valeurs passées par le trigger
au moment du lancement
(schedule, tumbling window, event...)
→ Même syntaxe que Parameters



Expressions

Functions ④

Outils pour manipuler/combiner les valeurs

Ex : concat(), utcnow(),
formatDateTime(), adddays()

Expression Builder : ⑤

combine tout ça dynamiquement

Exemple classique : @concat('bronze/',
pipeline().parameters.schema, '/',
formatDateTime(utcnow(),
'yyyy/MM/dd'))

The screenshot shows the Databricks Pipeline Expression Builder interface. The top navigation bar includes 'Home', 'Activities', 'Run', and 'View'. Below this is a toolbar with icons for 'Copy data', 'Dataflow', 'Notebook', 'Get metadata', and 'Lookup'. The main workspace is divided into two sections. The top section is a large empty box with a green plus icon in the top right corner. The bottom section is a 'Set variable' dialog box with a green header. It contains a blue 'x' icon, the text 'SetValue', and icons for trash, code editor, copy, and a green arrow. The right sidebar is titled 'Pipeline expression builder' and contains the text 'Add dynamic content below using any combination of expressions, functions and system variables.' Below this is a 'Clear contents' button and an 'Evaluate expression' button. The 'Evaluate expression' button is highlighted with a red circle ①. Below it are five tabs: 'Parameters' (highlighted with a red circle ②), 'System variables' (highlighted with a red circle ③), 'Trigger parameters' (highlighted with a red circle ④), 'Functions' (highlighted with a red circle ⑤), and an ellipsis. Below the tabs is a search bar with a magnifying glass icon and the text 'Search'. The bottom section of the interface is the 'Settings' tab, which has a 'Variable type' section with 'Pipeline variable' selected. It also has a 'Name' field with the value 'AnomaliesDetectees' and a 'Value' field.