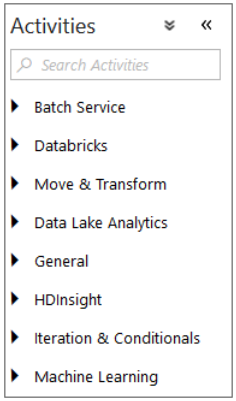
Chapitre 3
Activités et Data Flows en détail

1. Introduction

Ce chapitre décrit les activités existantes en dehors de tout contexte métier. Il ne s'agit pas d'une mise en situation de celles-ci, mais plus d'une énumération documentaire avec mise en avant de certaines options de configuration afin de les expliquer plus en détail. N'hésitez donc pas à le parcourir afin de prendre connaissance des différents types d'activités, ou alors à passer directement au chapitre suivant afin d'avoir des exemples d'utilisation plus concrets. Toutefois, chaque présentation est agrémentée des cas d'utilisation dans lesquels l'activité est communément utilisée.



2. Activité de flux de données

Les activités de flux de données sont au cœur d'Azure Data Factory, ce sont elles qui permettent de faire transiter les données d'un emplacement vers un autre. Les scénarios sont nombreux, qu'il s'agisse de copier des données depuis une base SQL Server On-Premise vers un stockage dans le Cloud, de faire des transformations entre plusieurs services Azure, ou même de copier des données depuis le Cloud vers un espace de stockage On-Premise. Il est important de comprendre que les activités de flux de données actuellement disponibles s'appuient sur deux mécanismes distincts.

L'activité Copy Data s'appuie sur les Integrations Runtimes afin d'effectuer la copie. Cela n'apporte pas une grande flexibilité et si les sources et destinations de données sont nombreuses, il n'est pas possible d'effectuer des transformations avancées. Toutefois, la mise en place est rapide, efficace et peu onéreuse et doit donc être privilégiée pour les opérations simples, particulièrement lors d'une utilisation dans un scénario hybride où les données peuvent être On-Premise.

L'activité Mapping Data Flow quant à elle, s'appuie sur le service DataBricks qui permet d'effectuer des transformations avancées et s'approche plus de ce qu'il était possible de réaliser avec SSIS. Cependant, les sources supportées sont moins nombreuses et nécessiteront souvent l'utilisation d'une activité **copy** en amont afin de stocker les données dans un stockage Azure. Cette activité est par ailleurs plus onéreuse qu'une activité de copie.

Remarque

En reprenant la philosophie propre aux architectures Cloud, Azure Data Factory fonctionne en proposant une série d'outils spécialisés. Il est donc nécessaire d'avoir une excellente connaissance des forces et faiblesses de chaque activité afin de choisir celle qui correspond au besoin en prenant en compte : la facilité de maintenance, le coût, les performances.

2.1 Copy Data

L'activité **Copy Data** est certainement la plus populaire d'Azure Data Factory. Elle permet d'effectuer une copie de données depuis une source vers une destination et pourra être configurée différemment en fonction du type de la source et de la destination.

Le code JSON permettant sa création est le suivant :

```
"activities":[
   {
       "name": "CopieActiviteModel",
       "type": "Copy",
       "inputs": [
           {
               "referenceName": "<Jeu de données source>",
               "type": "DatasetReference"
           }
       ],
       "outputs": [
           {
               "referenceName": "<Jeu de données destination>",
               "type": "DatasetReference"
           }
       ],
       "typeProperties": {
           "source": {
               "type": "<source type>",
               <properties>
           },
           "sink": {
               "type": "<destination type>"
               <properties>
           },
           "translator":
           {
               "type": "TabularTranslator",
               "columnMappings": "<column mapping>"
           },
           "dataIntegrationUnits": <number>,
           "parallelCopies": <number>,
           "enableStaging": true/false,
           "stagingSettings": {
```

```
                <properties>
            },
            "enableSkipIncompatibleRow": true/false,
            "redirectIncompatibleRowSettings": {
                <properties>
            }
        }
    }
]
```

– **type** : paramètre obligatoire. Dans le cas d'une activité de copie, la valeur doit être Copy.

– **inputs** : paramètre obligatoire référençant le jeu de données source. Même si ce paramètre laisse penser qu'il peut supporter plusieurs sources (via l'utilisation du pluriel et d'un tableau JSON), ce n'est pas le cas. Une activité de copie ne supporte pour l'instant de référencer qu'une seule source.

– **outputs** : paramètre obligatoire référençant le jeu de données de destination. Même si ce paramètre laisse penser qu'il peut supporter plusieurs destinations (via l'utilisation du pluriel et d'un tableau JSON), ce n'est pas le cas. Une activité de copie ne supporte pour l'instant de référencer qu'une seule destination.

– **typeProperties** : paramètre obligatoire. C'est ici que sont configurées les spécificités liées à cette instance de copie, telles que les spécificités liées aux sources et destinations, le mapping des colonnes, la gestion des rejets…

  – **source** : paramètre obligatoire, spécifie la façon dont les données sont récupérées au sein du jeu de données source. Par exemple, dans le cas d'une source de type Azure SQL Server, la configuration pourrait être la suivante :

```
"source": {
            "type": "SqlSource",
            "sqlReaderQuery": "SELECT * FROM MyTable"
        }
```

Remarque

Chaque élément de configuration dépend du type de source utilisé. Dans le cas d'une base de données, il est nécessaire de spécifier la requête SQL utilisée afin d'extraire les données.

– **sink** : paramètre obligatoire qui permet d'indiquer comment les données seront chargées dans la destination. Par exemple, dans le cas d'une destination de type Azure SQL Server, la configuration pourrait être la suivante :

```
"sink": {
                "type": "SqlSink",
                "writeBatchSize": 100000
            }
```

Remarque

Toutes les informations de configuration propres à chaque source ou destination peuvent être retrouvées depuis la documentation de Microsoft, ou depuis l'interface graphique du portail Azure Data Factory qui se chargera de générer le JSON.

– **translator** : paramètre facultatif, utilisé afin de faire le lien entre les colonnes source et destination. Dans le cas où les jeux de données sont de forme tabulaire (avec des colonnes et des lignes), par défaut le mapping est réalisé lorsque le nom des colonnes dans la source et la destination, sont identiques. Dans le cas où les noms de colonnes sont différents entre la source et la destination, il est possible de faire des liens de la façon suivante :

```
"translator":
        {
            "type": "TabularTranslator",
            "columnMappings":
            {
                "SourceNomCol1": "DestNomCol1",
                "SourceNomCol2": " DestNomCol2",
            }
        }
```

Remarque

*Notez ici qu'il s'agit de la propriété **columnMappings** qui est utilisée.*

Il est aussi possible de réaliser des opérations plus complexes telles qu'un mapping entre une source de données MongoDB (Bson) vers une destination Azure SQLDB. Dans ce cas, il est nécessaire de réaliser un mapping de schéma.

```
"translator": {
            "type": "TabularTranslator",
            "schemaMapping": {
                "orderNumber": "$.number",
                "orderDate": "$.date",
                "order_pd": "prod",
                "order_price": "price",
                "city": " $.city[0].name"
            },
            "collectionReference":  "$.orders"
        }
```

– **dataIntegrationUnits** : paramètre facultatif, indique le nombre d'unités de traitement utilisées par l'Azure Integration Runtimes afin de copier les données. Attention, cela a un impact sur le coût d'utilisation de l'Integration Runtimes.

– **parallelCopies** : paramètre facultatif, détermine le niveau de parallélisation utilisé lors de la copie. Attention à bien mesurer le gain d'un changement sur cette propriété, les valeurs par défaut étant spécifiées dynamiquement par Microsoft avec la règle suivante :

Scénario	Valeur par défaut
Copie de fichiers	Calculée en fonction du nombre de DataIntegrationUnits dans le cas d'un Azure IR ou de la taille de l'IR Auto-Hébergé
Copie vers Azure Table Storage	4
Autres sources et destinations	1