Azure Data Factory

Hands-on lab step-by-step

April 2019

Information in this document, including URL and other Internet Web site references, is subject to change without notice. Unless otherwise noted, the example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, e-mail address, logo, person, place or event is intended or should be inferred. Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

The names of manufacturers, products, or URLs are provided for informational purposes only and Microsoft makes no representations and warranties, either expressed, implied, or statutory, regarding these manufacturers or the use of the products with any Microsoft technologies. The inclusion of a manufacturer or product does not imply endorsement of Microsoft of the manufacturer or product. Links may be provided to third party sites. Such sites are not under the control of Microsoft and Microsoft is not responsible for the contents of any linked site or any link contained in a linked site, or any changes or updates to such sites. Microsoft is not responsible for webcasting or any other form of transmission received from any linked site. Microsoft is providing these links to you only as a convenience, and the inclusion of any link does not imply endorsement of Microsoft of the site or the products contained therein.

© 2018 Microsoft Corporation. All rights reserved.

Microsoft and the trademarks listed at <https://www.microsoft.com/en-us/legal/intellectualproperty/Trademarks/Usage/General.aspx> are trademarks of the Microsoft group of companies. All other trademarks are property of their respective owners.

Contents

[Azure Data Factory hands-on lab step-by-step 1](#_Toc5661800)

[Abstract and learning objectives 1](#_Toc5661801)

[Overview 1](#_Toc5661802)

[Solution architecture 2](#_Toc5661803)

[Requirements 2](#_Toc5661804)

[Before the hands-on lab 3](#_Toc5661805)

[Task 1: Deploy Azure Data Factory and Storage Accounts to Azure 3](#_Toc5661806)

[Task2: Replicate the WWI Historical data Blob storage using SAS URI 5](#_Toc5661807)

[Exercise 1: 6](#_Toc5661808)

[Task 1: Create Azure Data Factory Integration Runtime. 6](#_Toc5661809)

[Task 2: Create a new ADF pipeline 8](#_Toc5661810)

[Task 3: Create the Sink dataset as an Azure SQL DB 11](#_Toc5661811)

[Task 4: Create a stored procedure to perform ETL in SQL DB 15](#_Toc5661812)

[Task 5: Add a stored procedure activity to the pipeline 15](#_Toc5661813)

[Task 6: create a pipeline to run daily load 17](#_Toc5661814)

[Task 7: create a pipeline to iteratively perform historical load 20](#_Toc5661815)

[After the hands-on lab 23](#_Toc5661816)

[Task 1: Delete resource group 23](#_Toc5661817)

[Appendix 23](#_Toc5661818)

[Customer Table DDL: 23](#_Toc5661819)

[Orders Table DDL: 24](#_Toc5661820)

[orderlines Table DDL: 25](#_Toc5661821)

# Azure Data Factory hands-on lab step-by-step

## Abstract and learning objectives

In this workshop, you will deploy a Azure Data Factory to perform ETL operation using Azure Blob storage, Azure SQL DB and Azure Databricks(Optional).

By attending this workshop, you will be better able to build a complete Azure data factory ETL pipeline. In addition, you will learn to:

* Deploy Azure Data Factory including an Integration Runtime.
* Create Blob Storage and Azure SQLDB Linked Services.
* Create ADF parameterized pipeline.
* Perform ETL using Azure SQLDB Store Procedures
* Perform ELT using Azure Databricks Notebooks and Azure Data Lake Storage Gen2. (Optional)

This hands-on lab is designed to provide exposure to many of Microsoft’s transformative line of business applications built using Microsoft data and advanced analytics. The goal is to show an end-to-end solution, leveraging many of these technologies, but not necessarily doing work in every component possible. The lab architecture is below and includes:

* Azure Data Factory (ADF)
* Azure Storage
* Azure Data Lake Storage Gen2
* Azure Databricks

## Overview

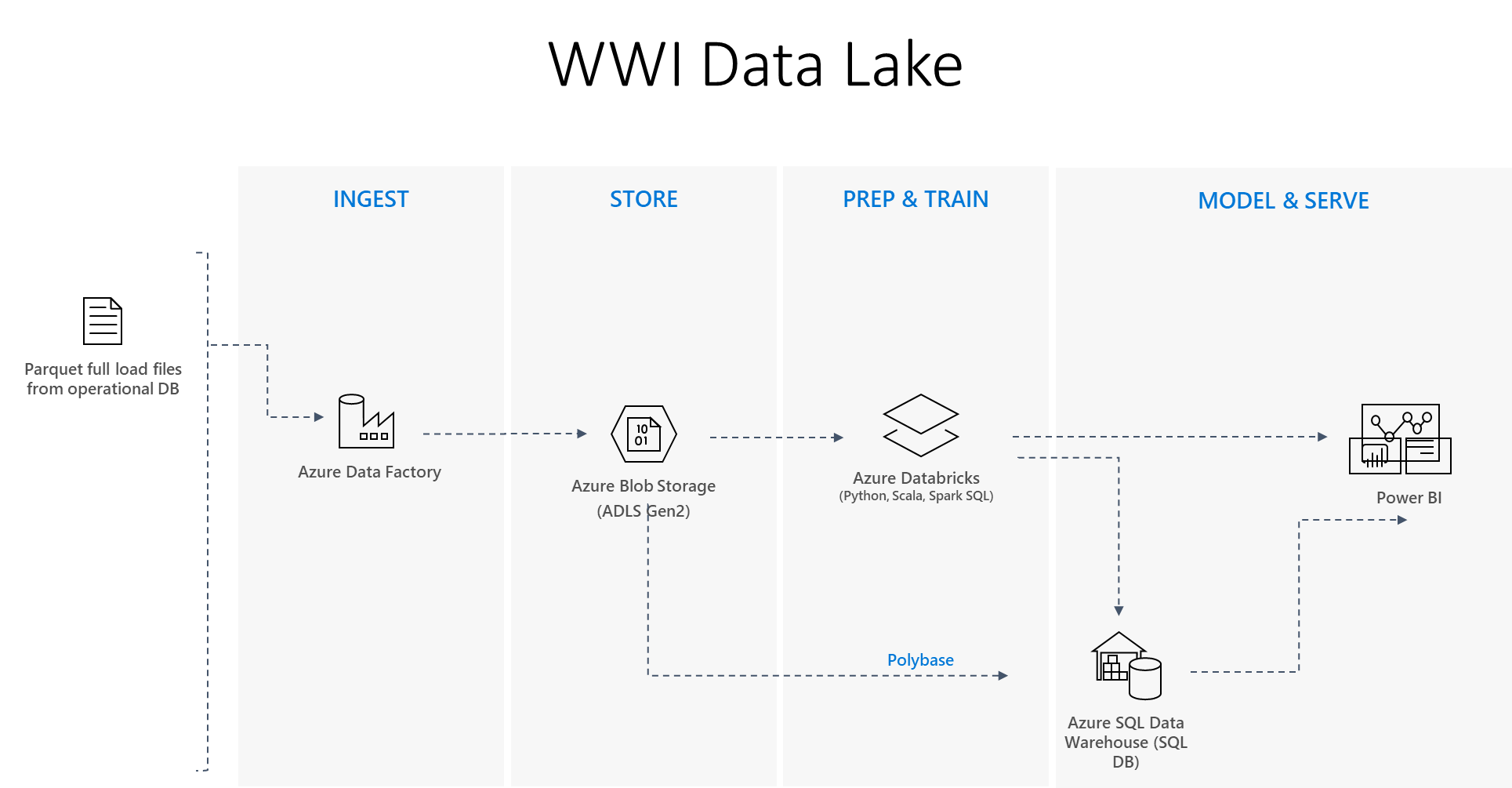
WorldWide Imorters (WWI) imports all a wide range of products which then resells to retailers and public directly. In an increasingly crowded market, they are always looking for ways to differentiate themselves, and provide added value to their customers.

They are looking to pilot a data warehouse to provide additional information useful to the their internal sales and marketing agents. They want to enable their agents to perform AS-IS and AS-WAS analysis in order to price the items more accurately and predict the product demand at different times during the year.

In this hands-on lab, attendees will build an end-to-end solution to build a data warehouse using data lake methodology.

## Solution architecture

Below is a diagram of the solution architecture you will build in this lab. Please study this carefully so you understand the whole of the solution as you are working on the various components.



The solution begins with loading their daily operational data into blob storage using Azure Data Factory (ADF). By setting up a pipeline containing a copy activity configured to copy date partitioned source data. By using a parameterized pipeline, they could pull their historical information, as well as ingest any future data, into Azure blob storage through a scheduled, and continuously running pipeline. Source data will be delivered from their on-prem operational database in form of Parquet files to an Azure Blob Storage container. Files are placed in a staging container and formatted in <domain>\_<date\_of\_retrieval>.parquet and placed in virtual directories in the container as <domain>.

## Requirements

1. Microsoft Azure subscription must be pay-as-you-go or MSDN
   1. Trial subscriptions will not work if using Azure DataBricks for ELT

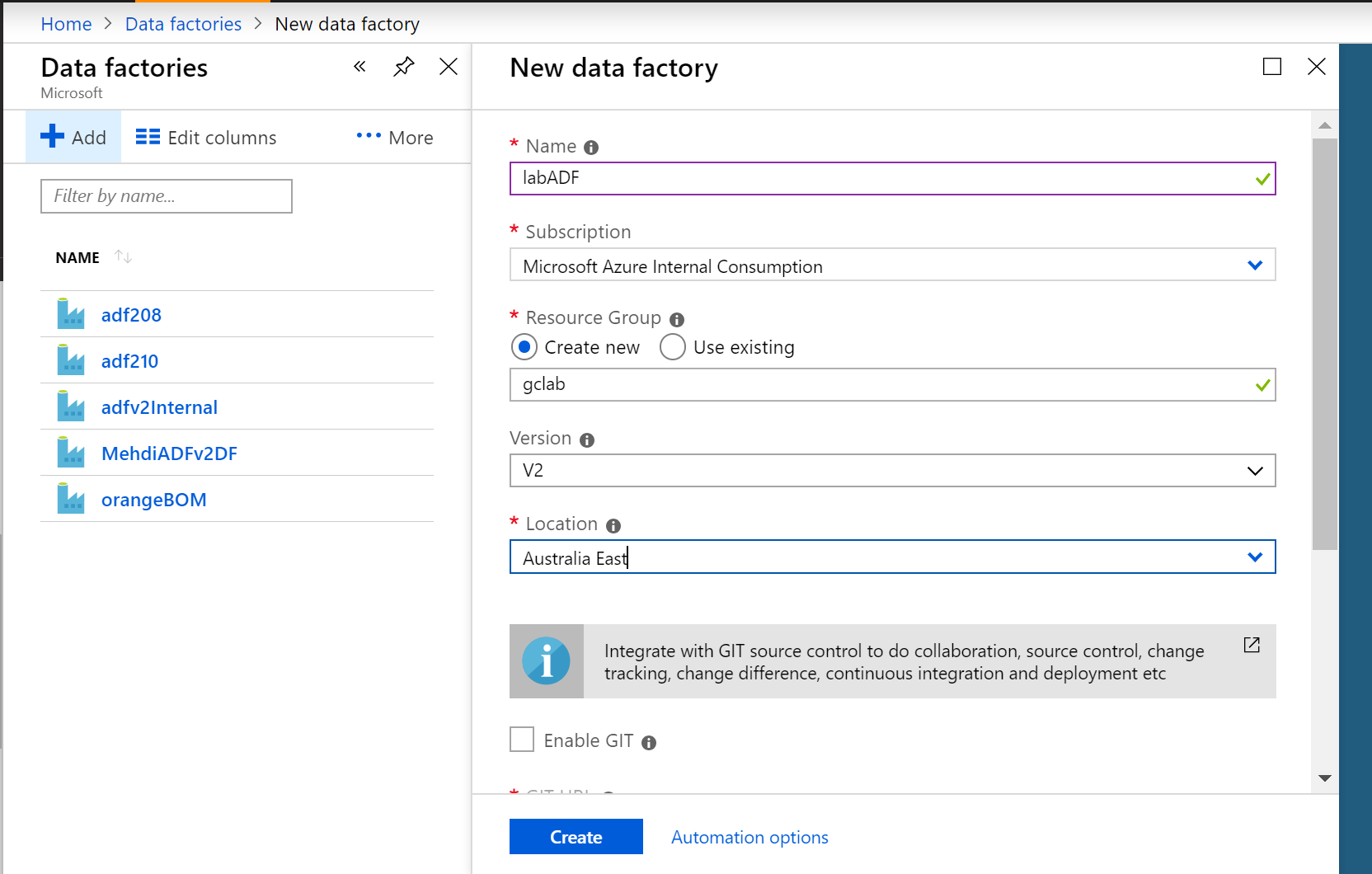
## Before the hands-on lab

Duration: 20 minutes

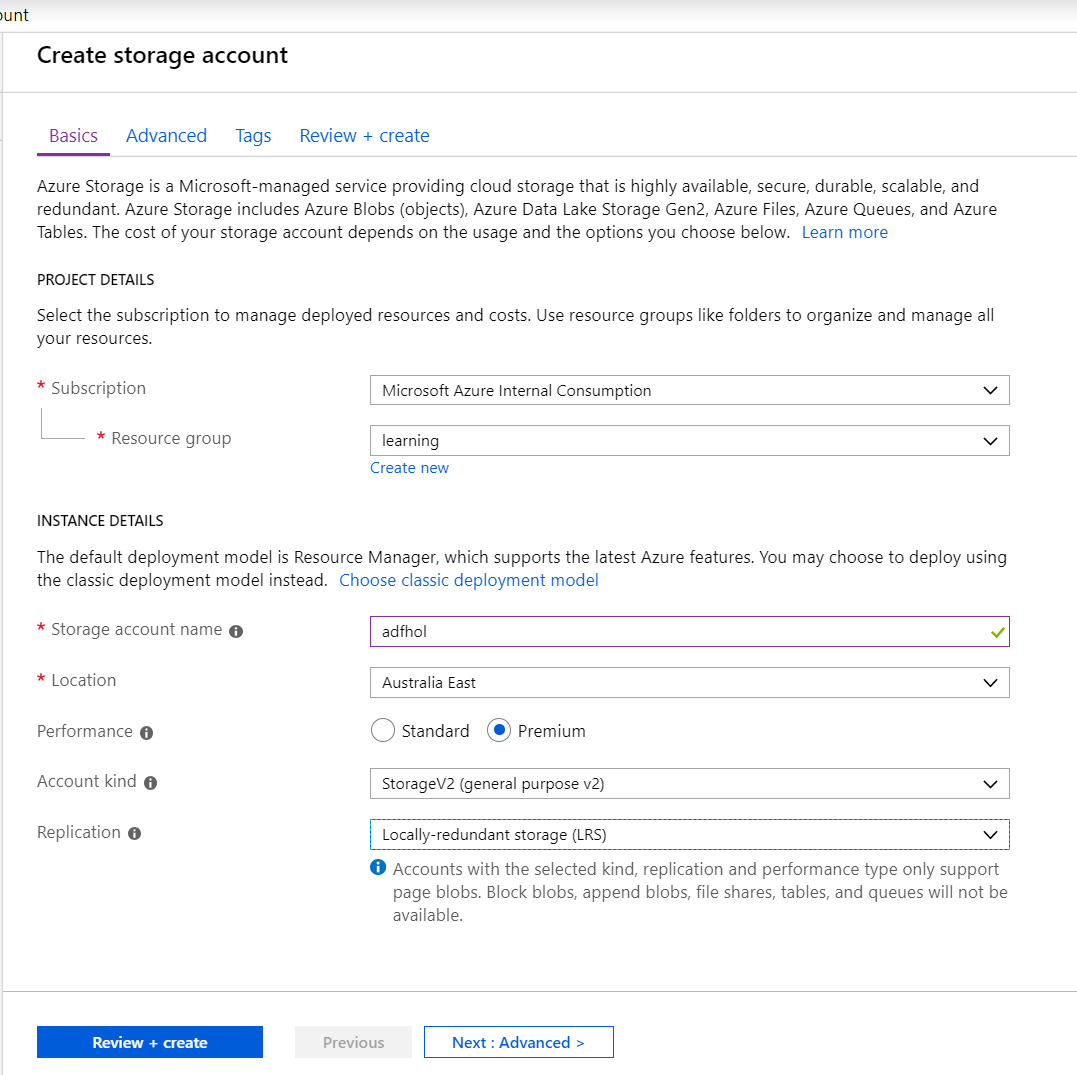
In this exercise, you will set up your environment for use in the rest of the hands-on lab. You should follow all the steps provided in the Before the Hands-on Lab section to prepare your environment *before* attending the hands-on lab.

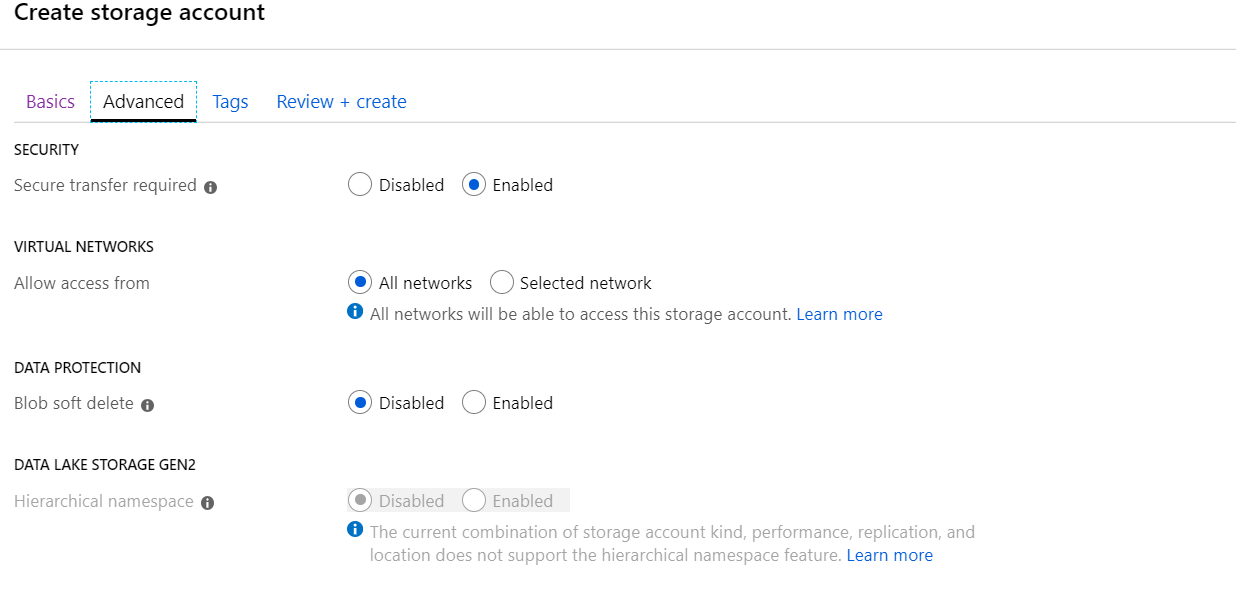
#### Task 1: Deploy Azure Data Factory and Storage Accounts to Azure

1. Deploy and Azure Data Factory in the region of your choice.



1. Deploy an Azure Storage Account as below

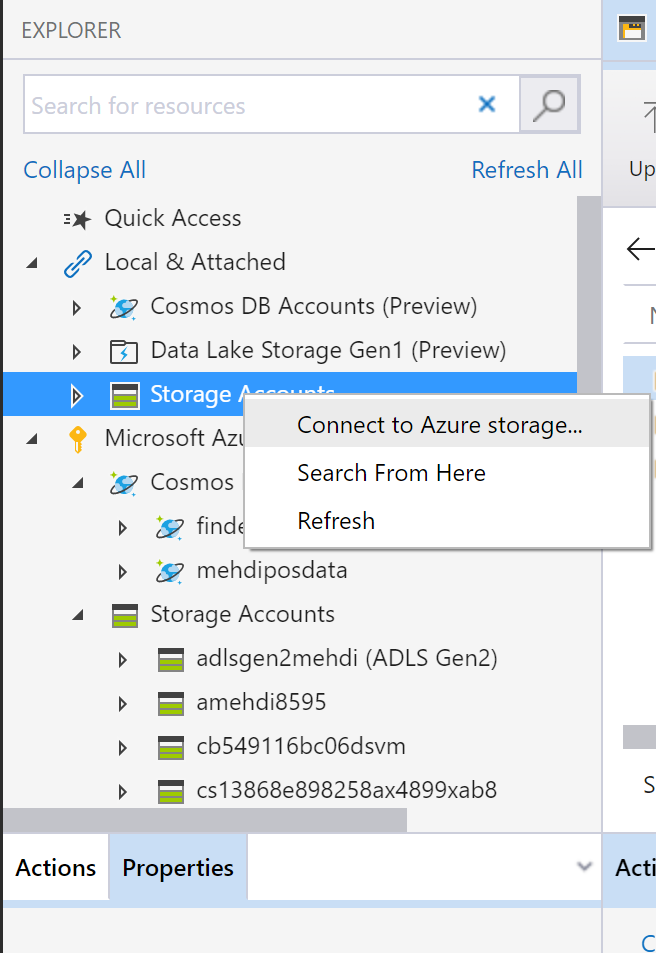
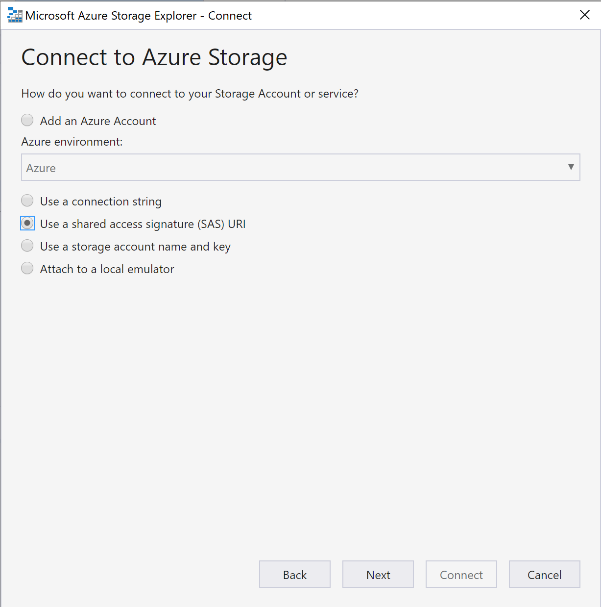
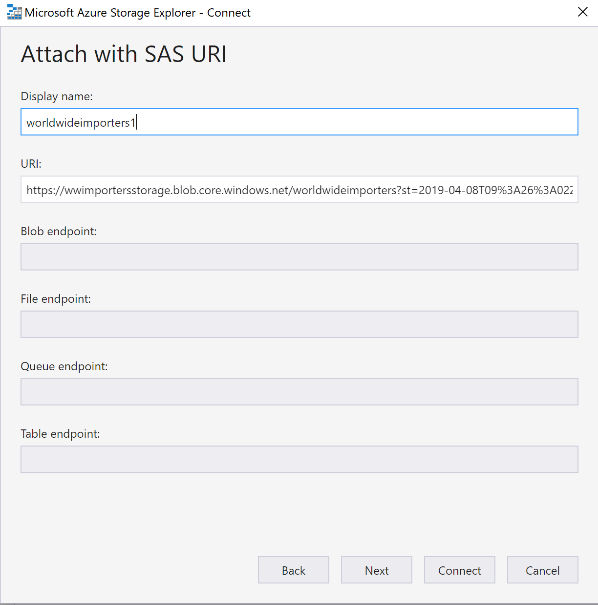


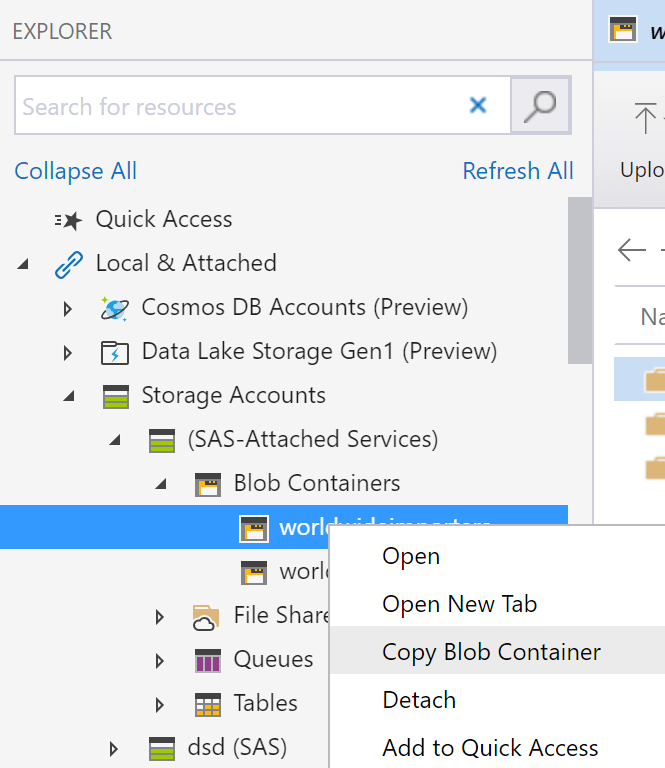


#### Task2: Replicate the WWI Historical data Blob storage using SAS URI

1. Install Azure Storage Explorer and login to your Azure subscription
2. Connect to the following Azure Blob container using SAS URI and copy it to your own Storage Account

<https://wwimportersstorage.blob.core.windows.net/worldwideimporters?st=2019-04-08T09%3A26%3A02Z&se=2019-04-14T09%3A26%3A00Z&sp=rl&sv=2018-03-28&sr=c&sig=7hFW6xHBQ4WbrZzkNvfmqjUNrrDTUzf1dc%2Ff7ErmaTs%3D>







#### 

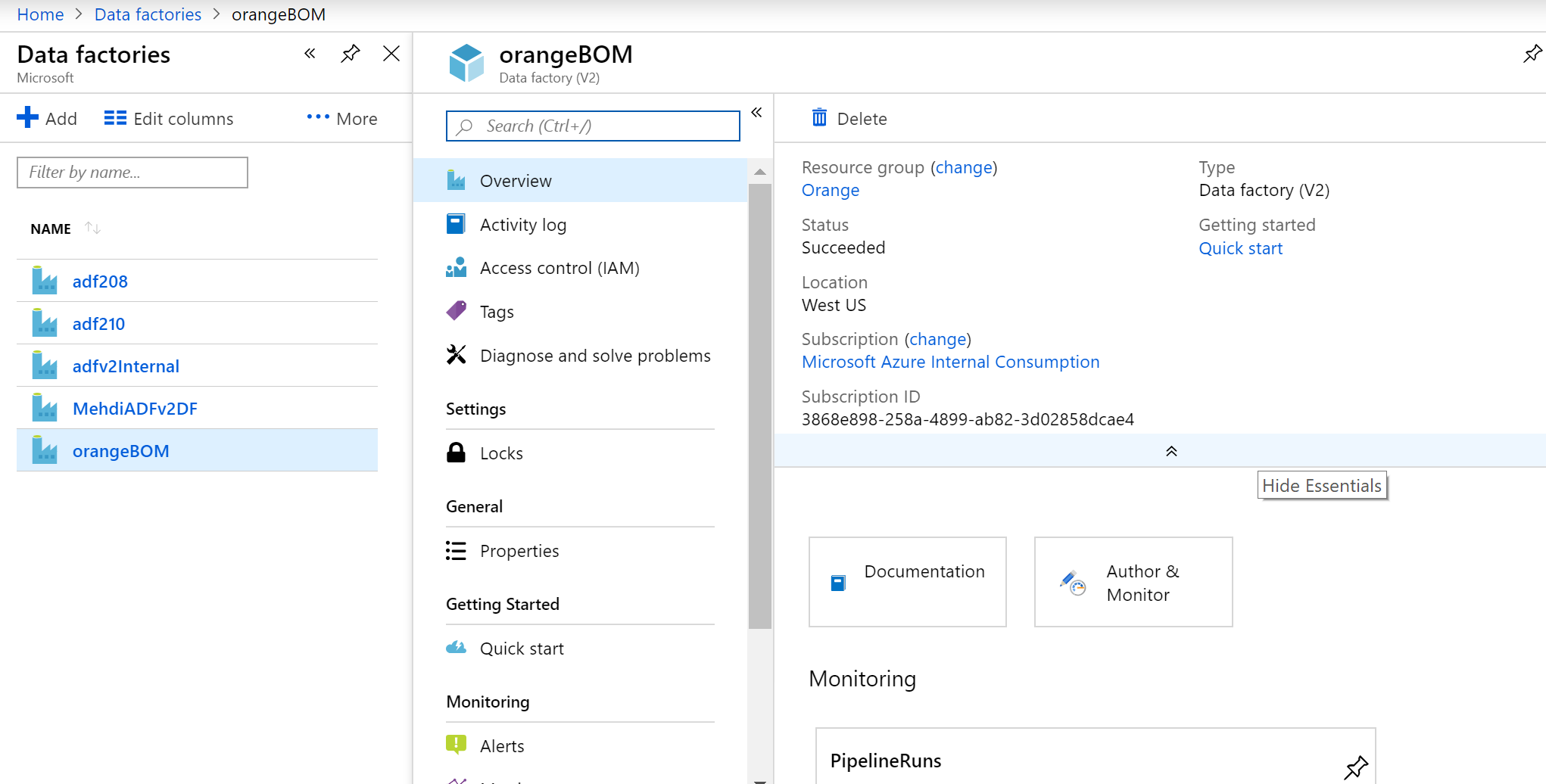
## Exercise 1:

Duration: 120 minutes

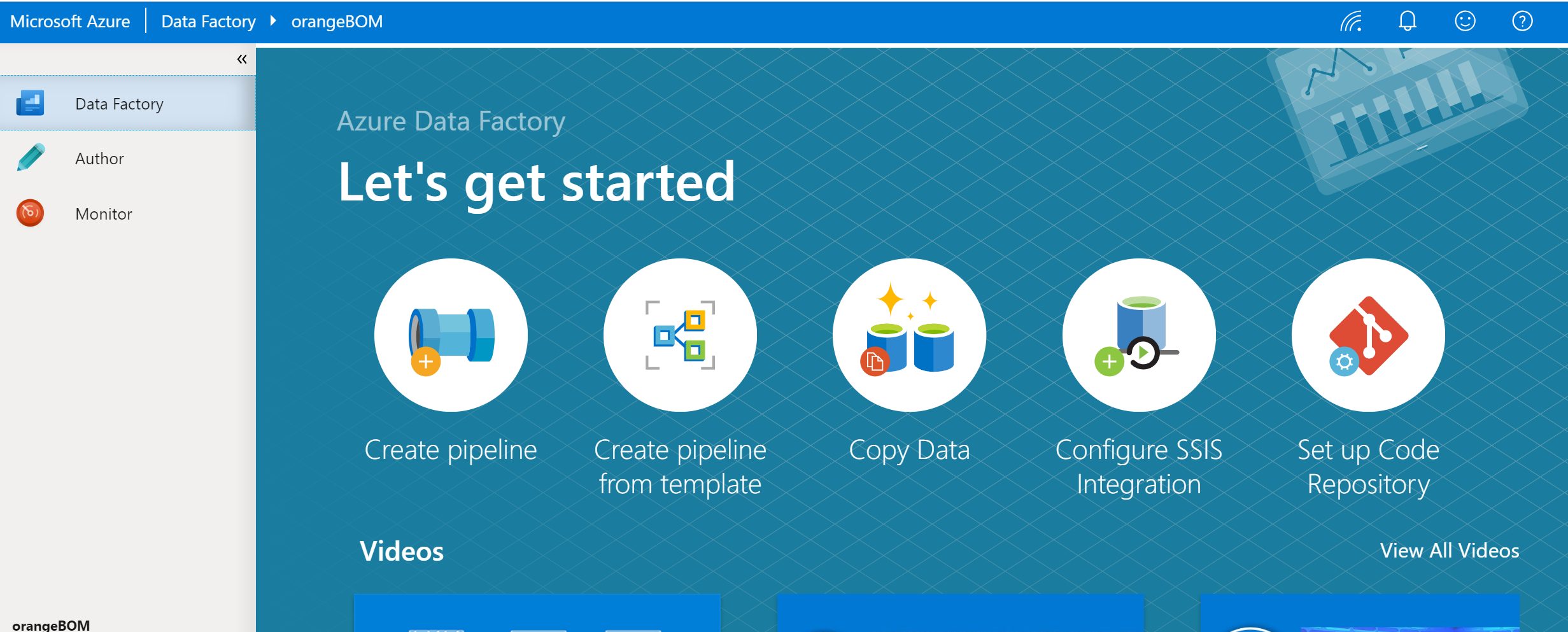
In this exercise, attendees will create multiple Azure data factory pipelines and related component including ADF IR, LinkedServices and Datasets.

#### Task 1: Create Azure Data Factory Integration Runtime.

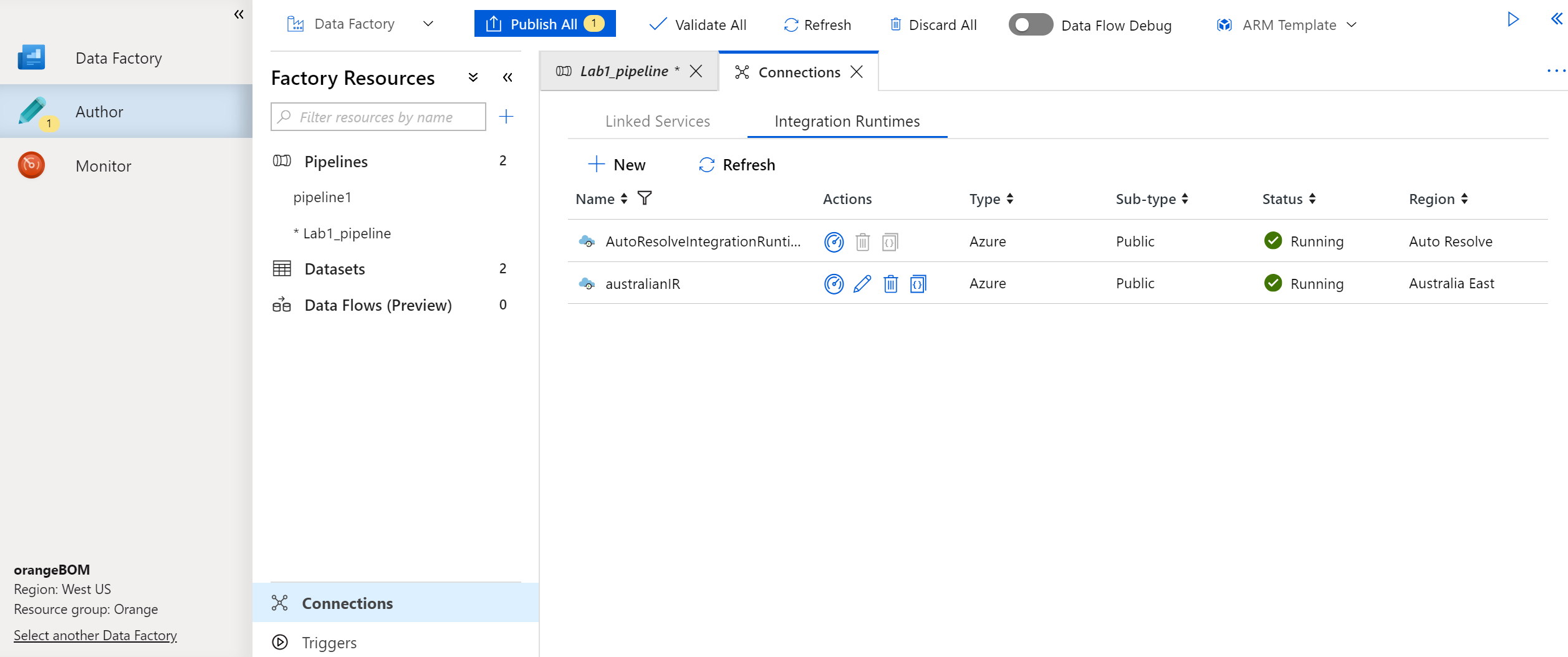
1. Open Azure Data Factory resource in Azure Portal



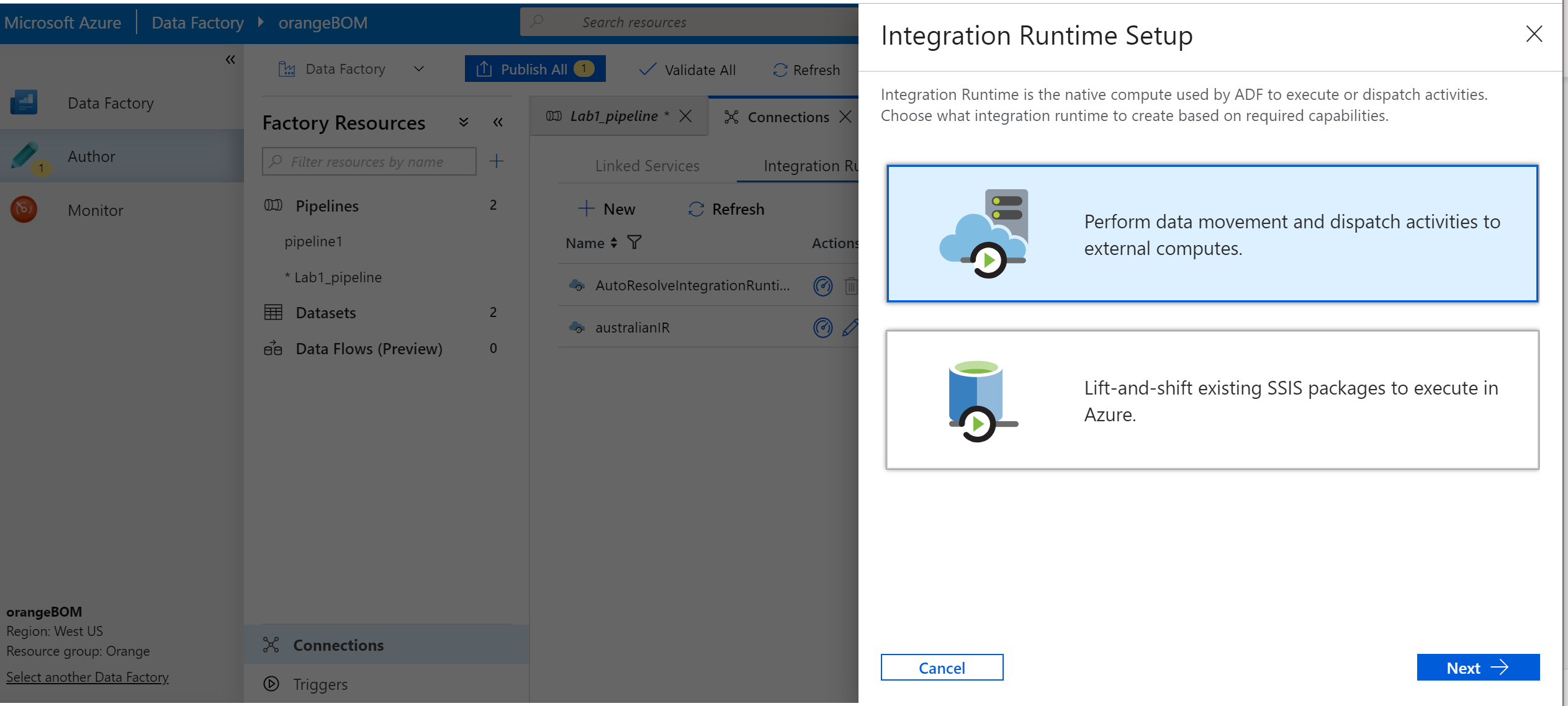
1. Click on Author and Monitor and it should be taking you to ADF UI like below. This is the main ADF UI. On the left hand side there is Author and Monitor tabs. click on author tab to start building the first pipeline.



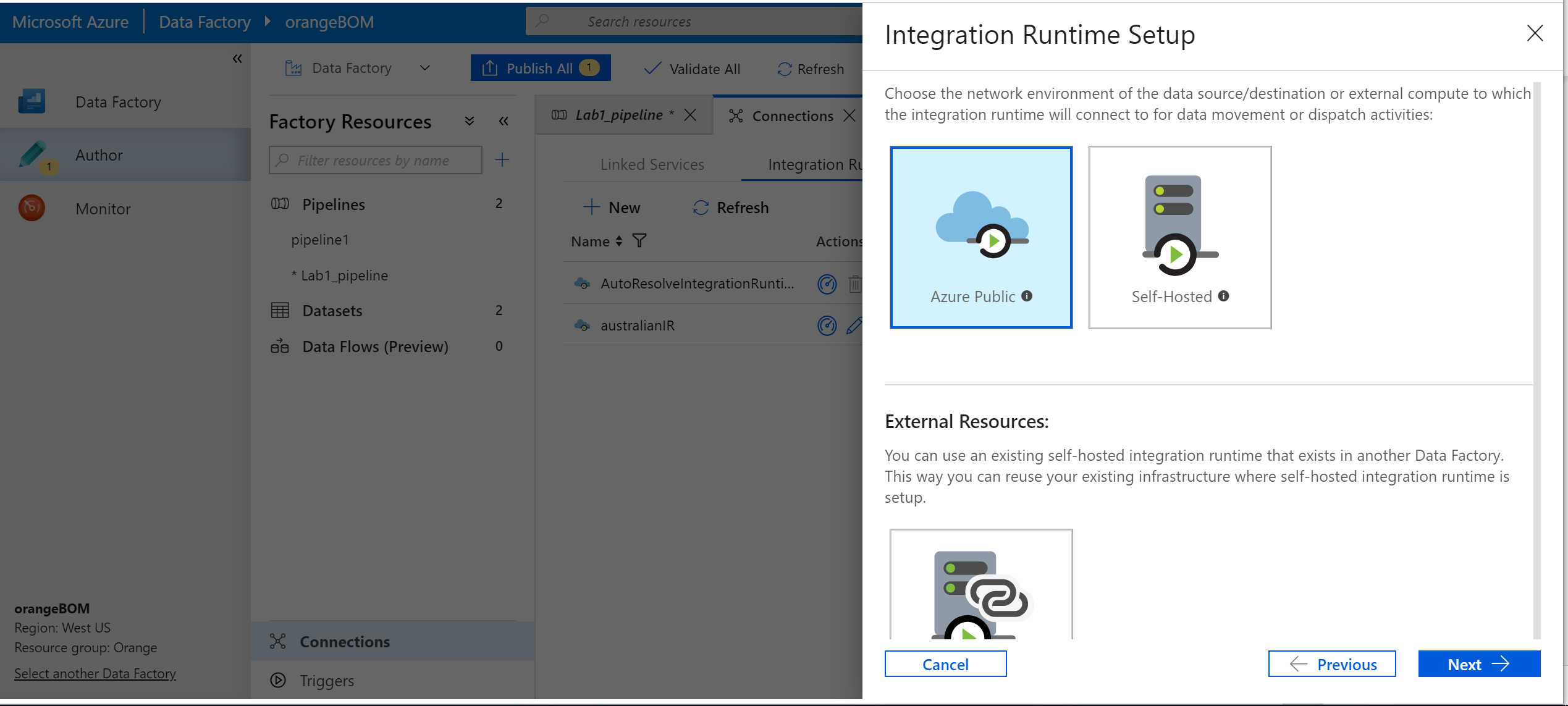
1. Create a new Integration run time in a particular Azure Region: go to Author tab🡪 Connections (at the bottom of the screen) 🡪 click New



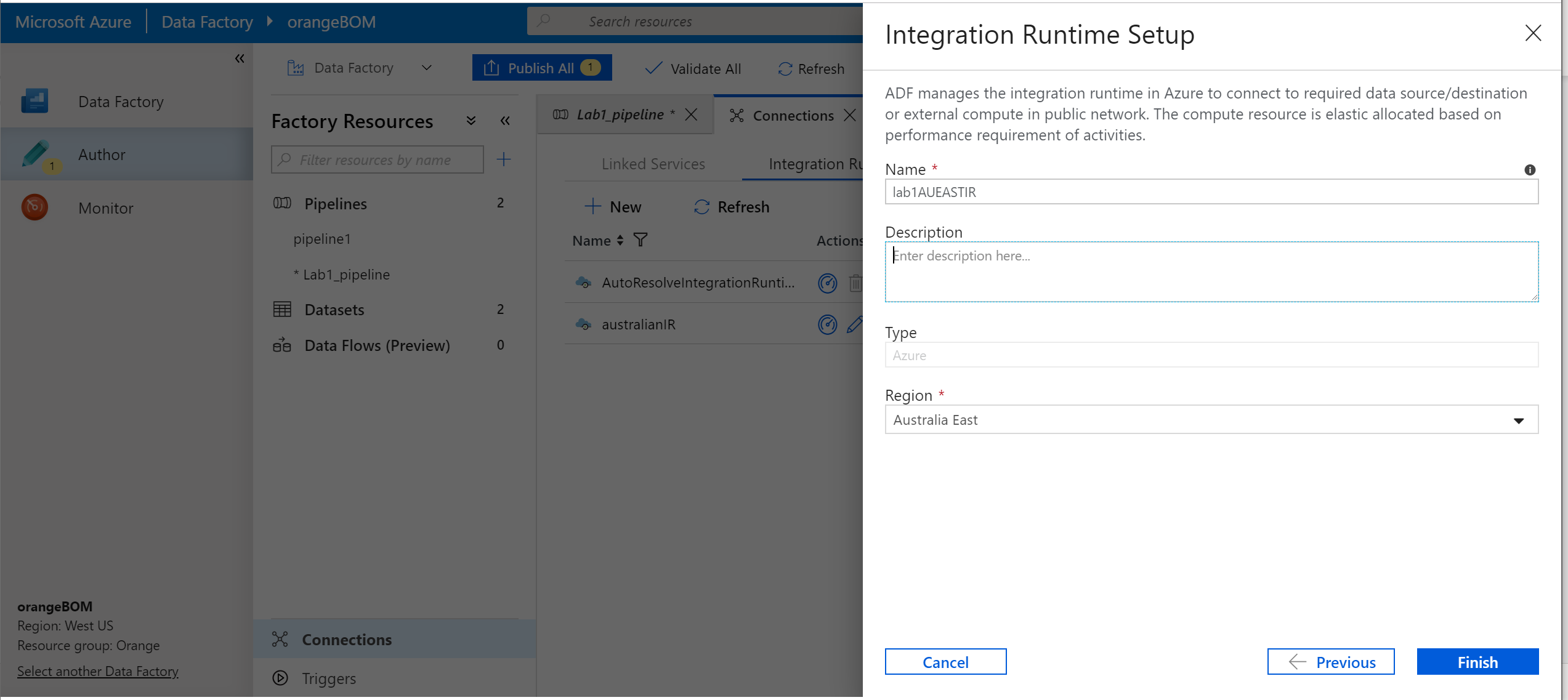
Select “Perform data movement and dispatch activities to external computes”



Choose “Azure Public”

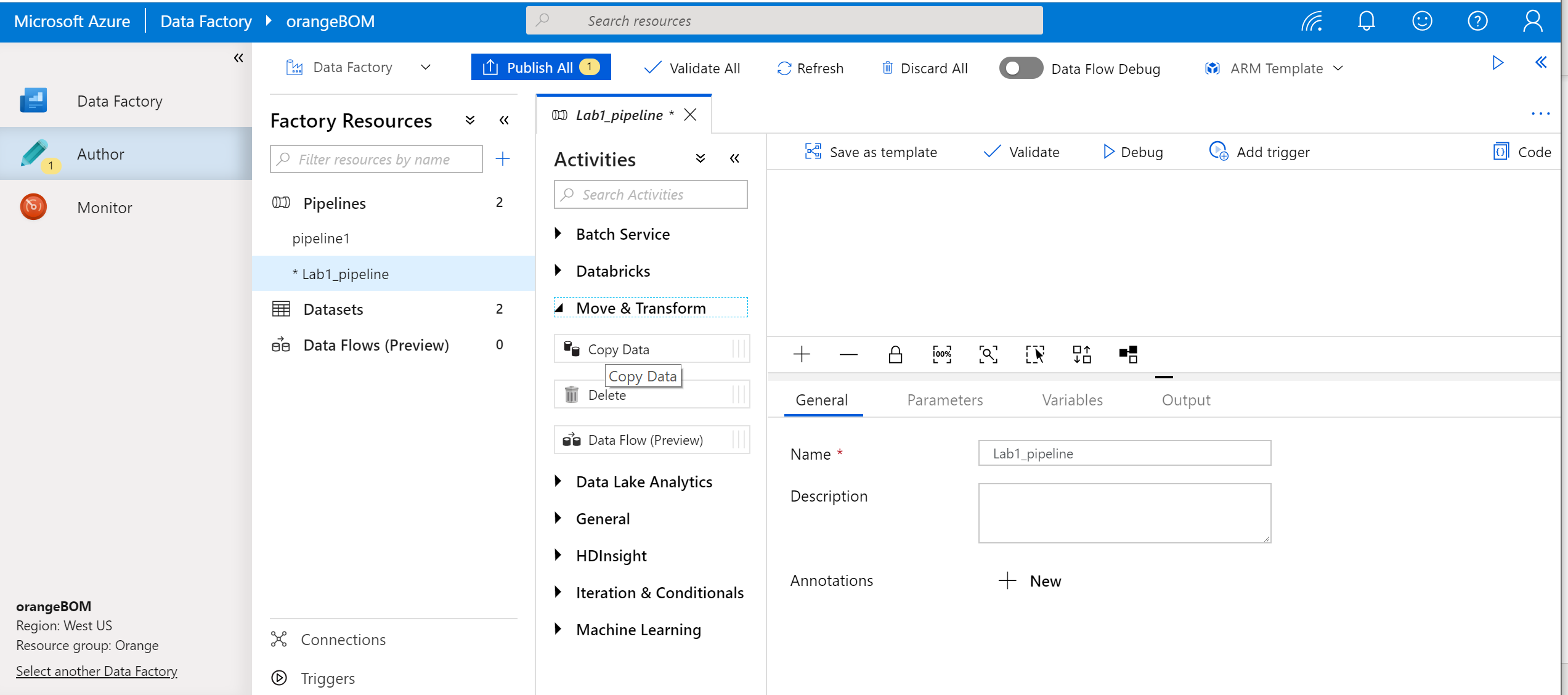


Give your new IR a name and Choose the preferred region

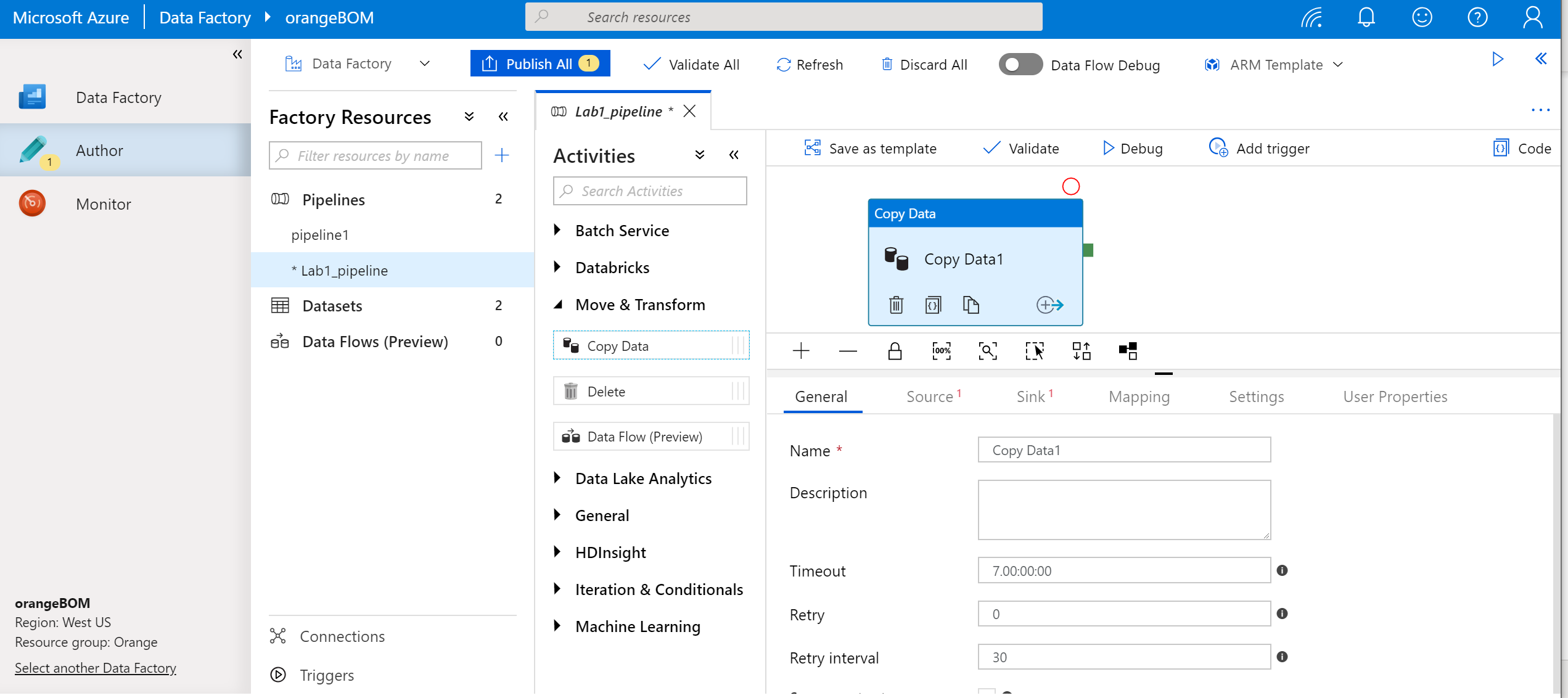


#### Task 2: Create a new ADF pipeline

1. Create a new pipeline, by selecting the plus sign under “Factory Resources” and choosing “Pipeline”. Once created you have a new canvas to place ADF activities and set them up.

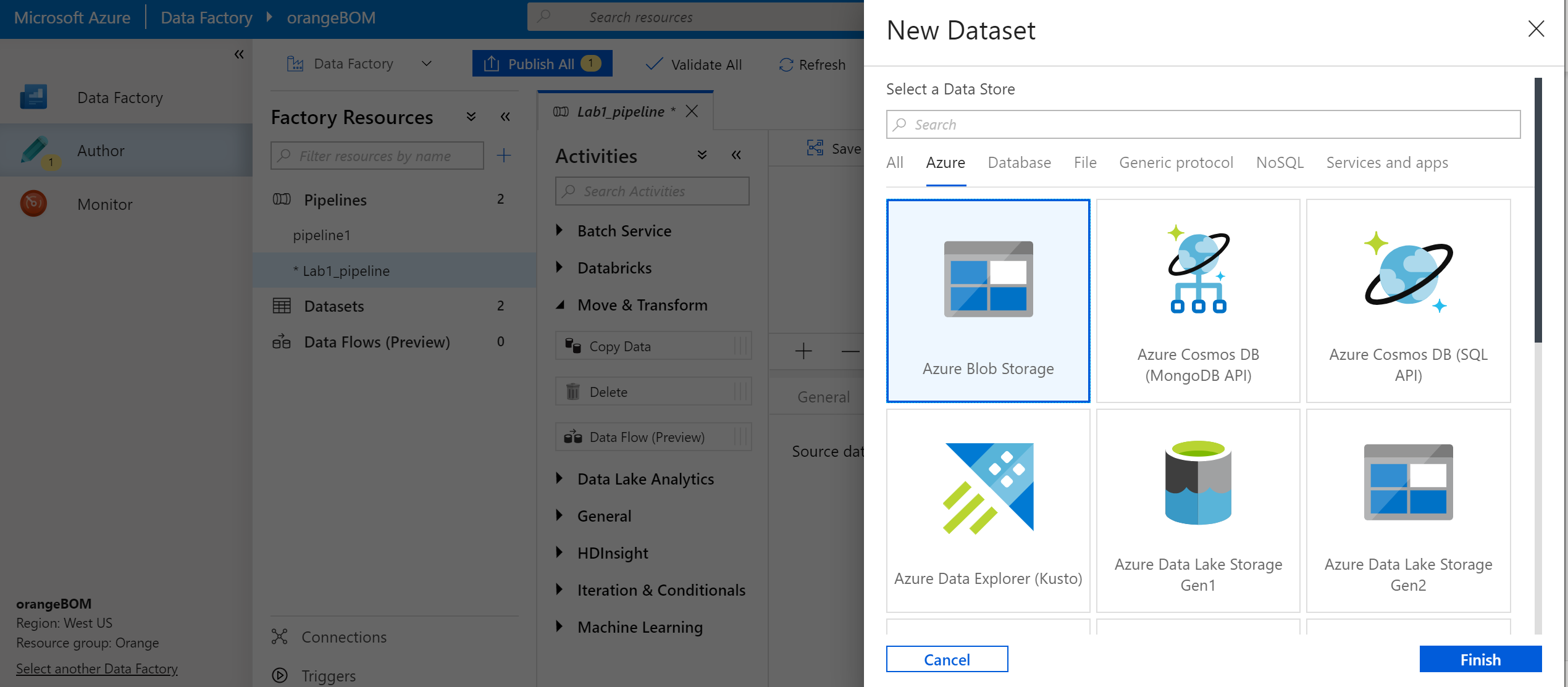


1. Place your first ADF Activity (Copy data) from Move and Transform menu on the canvas. And Rename it to something meaningful.

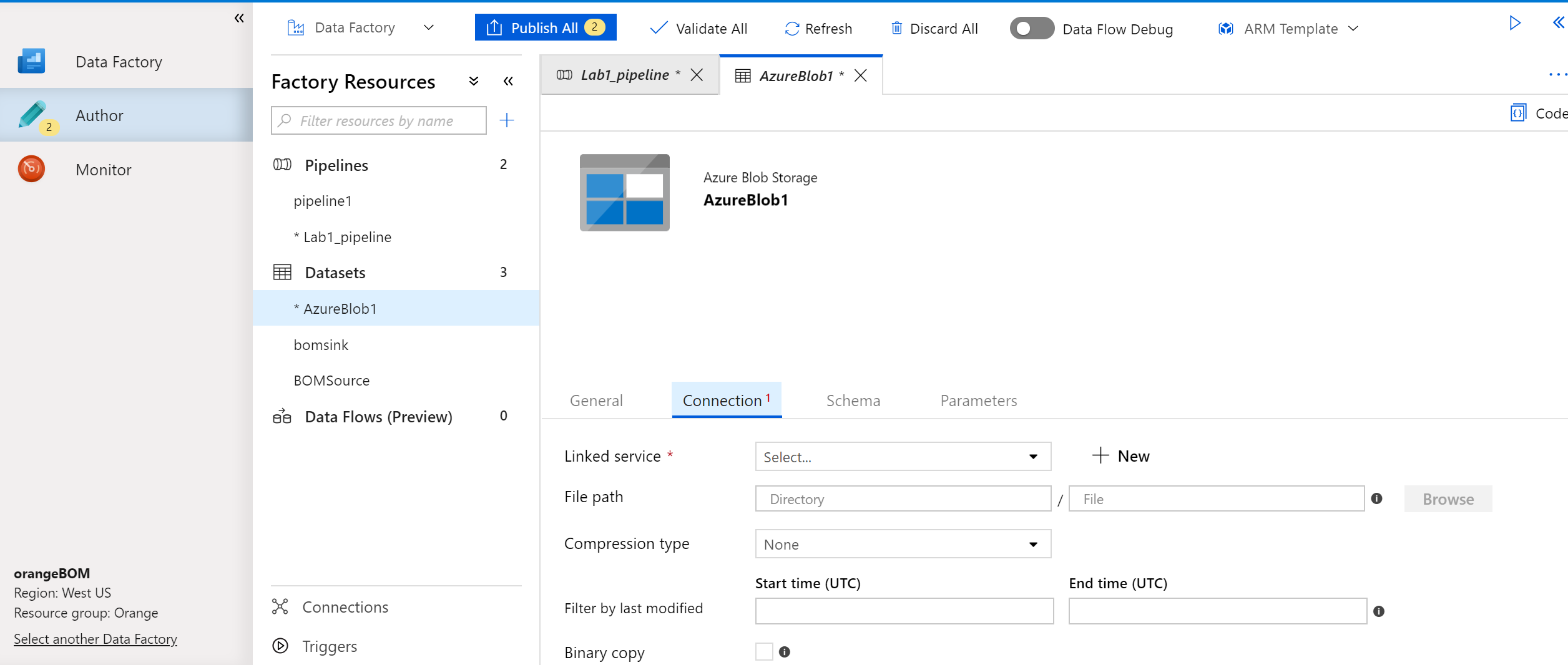


1. Create the Source dataset:

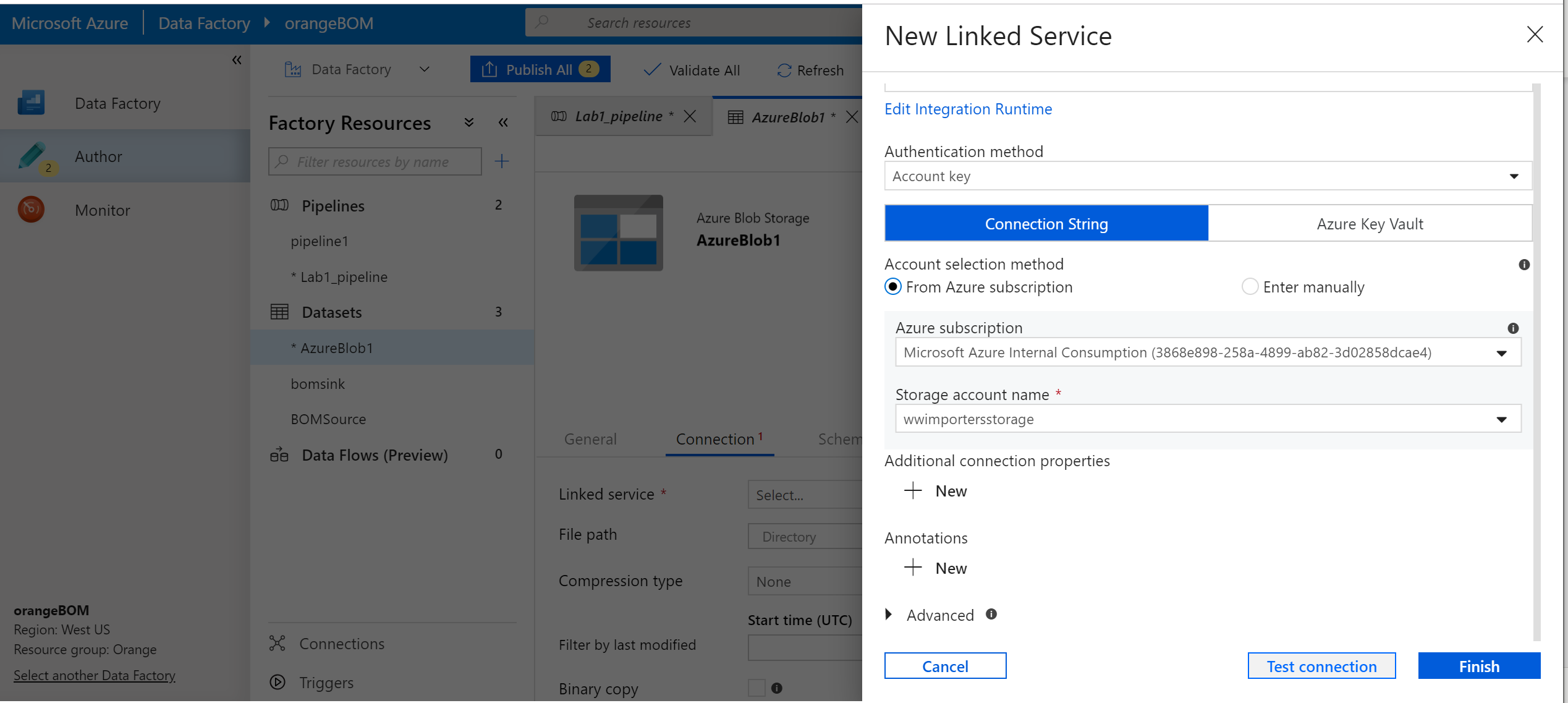
Select source tab in the copy activity and select Blob Storage. This will take your new blob storage source.



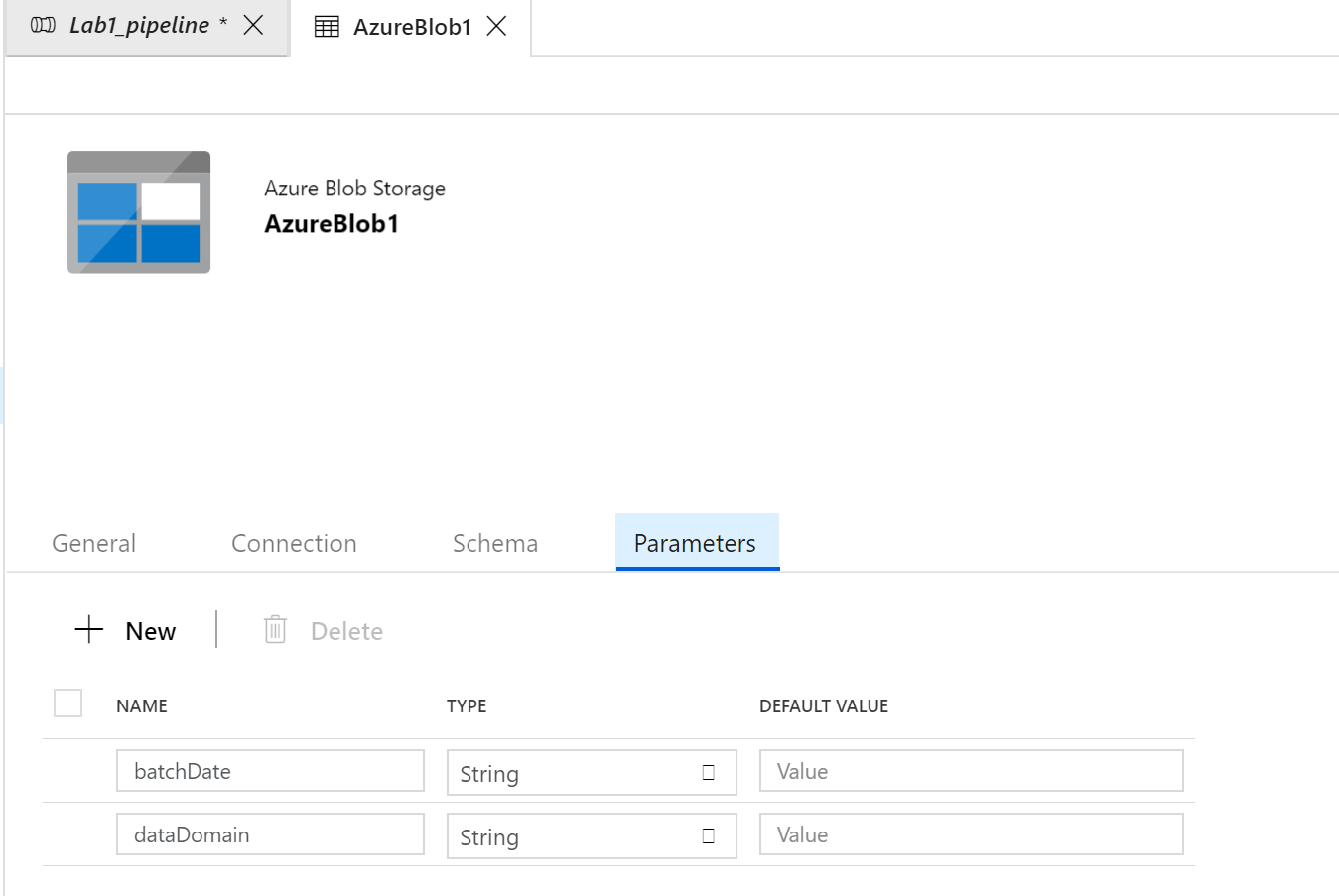
1. Select new under connection tab to create a new Linked Service for the blob source to connect to Azure blob storage.



1. Give your new “Linked Service” a name and set it up to the Storage account you created in previous steps.



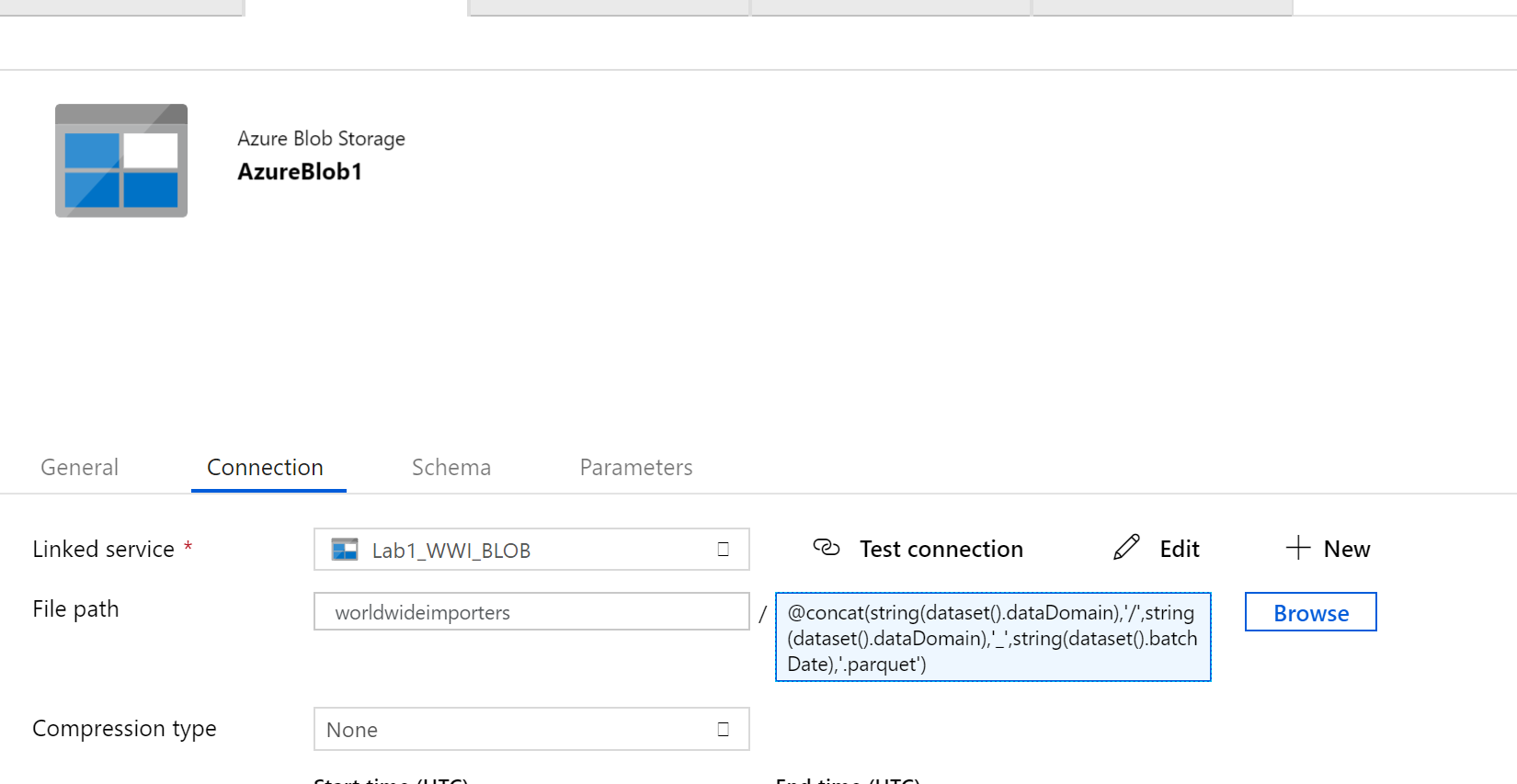
1. Now we need to create some parameters in the source data set. Create **batchDate** and **dataDomain** dataset parameters as String type.



1. Now in the connection tab of the dataset set the path using the created parameters.

Set the container to : “worldwideimporters” and the file path to:

@concat(string(dataset().dataDomain),'/',string(dataset().dataDomain),'\_',string(dataset().batchDate),'.parquet')



1. At the bottom of the page change “File Format” to “Parquet



**Note:** Go through the expression language used in the file path provided and try to understand how it was built.

1. Now we need to go back to our pipeline and fill in the value for the parameters we used in the data source



For batchDate: @pipeline().parameters.batchDate

For dataDomain: @pipeline().parameters.dataDomain

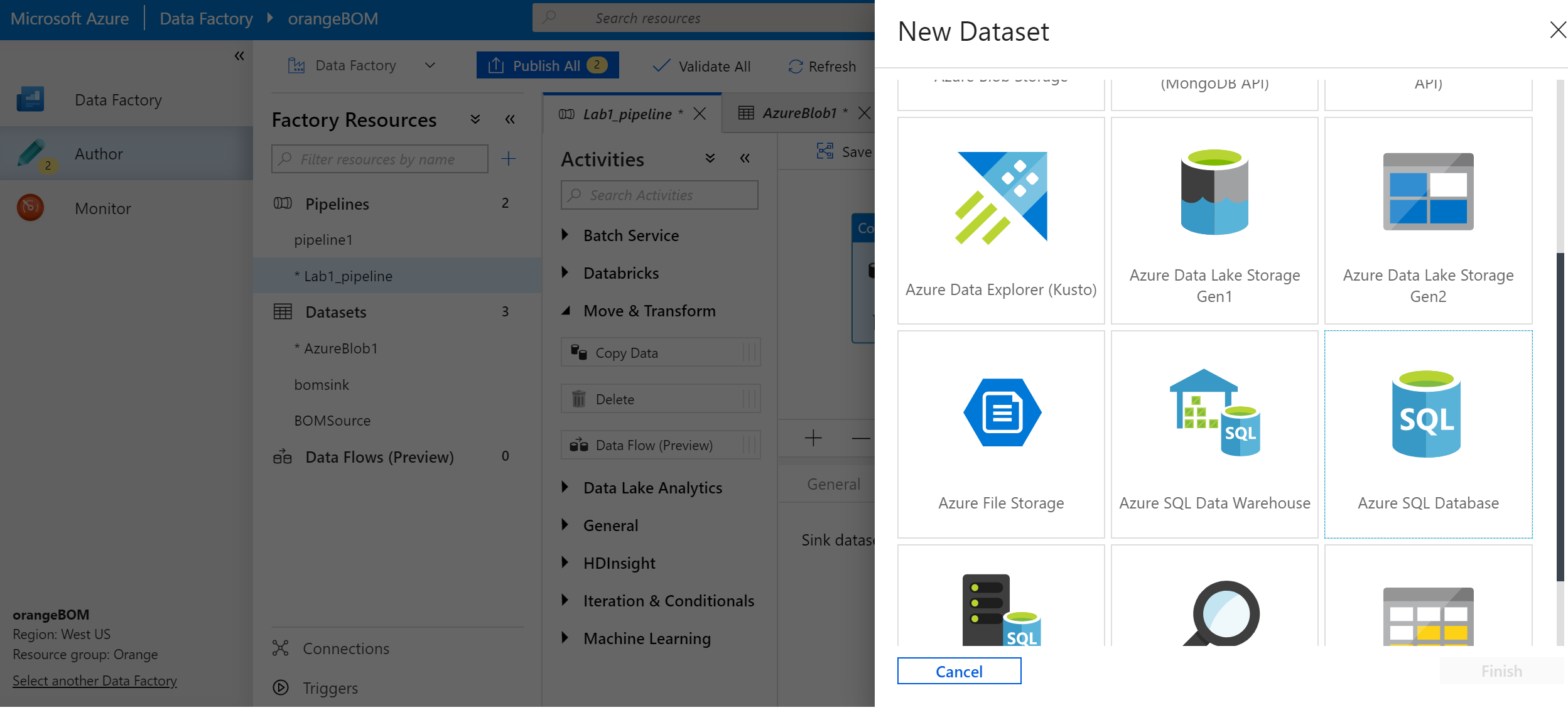
**Note:** What is the different between the pipeline parameters and dataset parameters?!

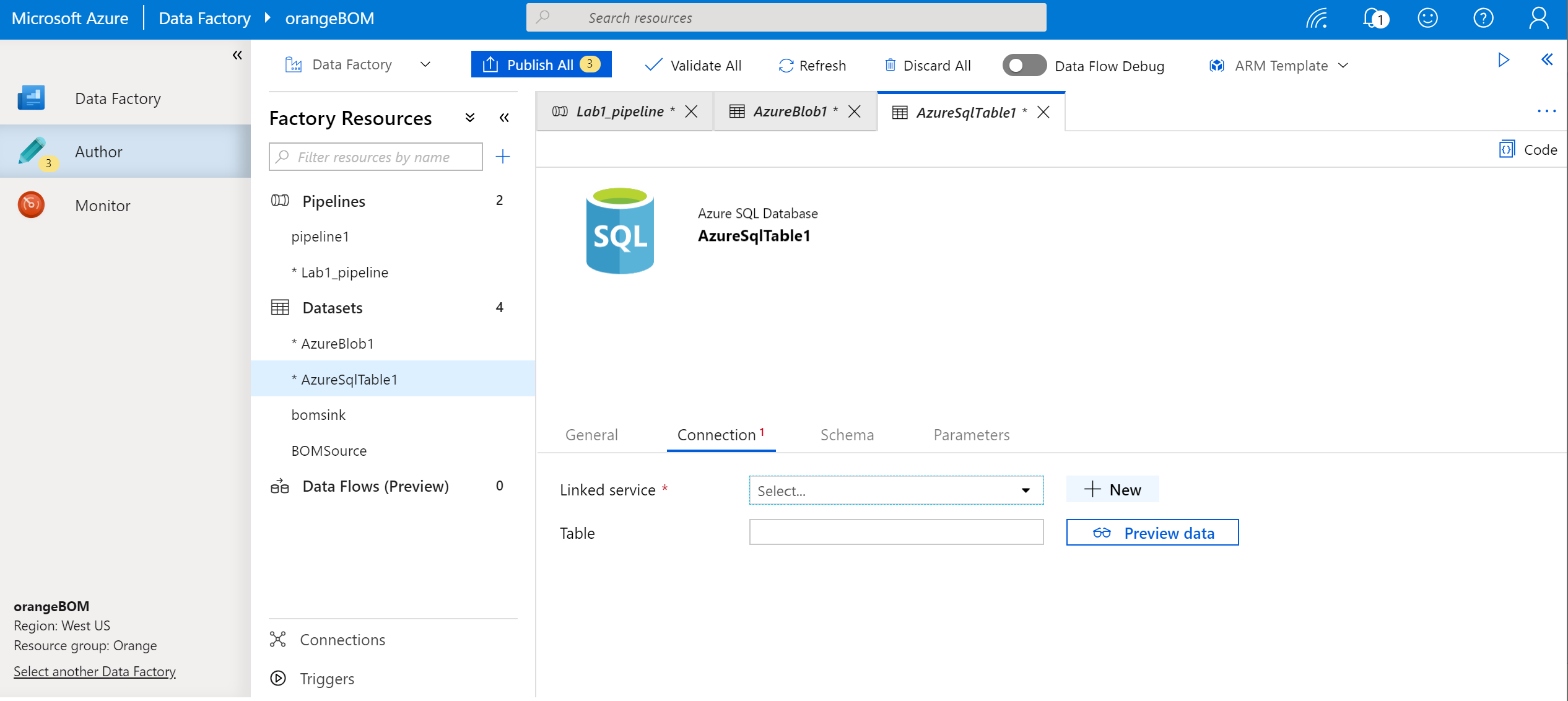
**Note:** Are you getting an error for the created expressions? Do you need to create pipeline parameters?

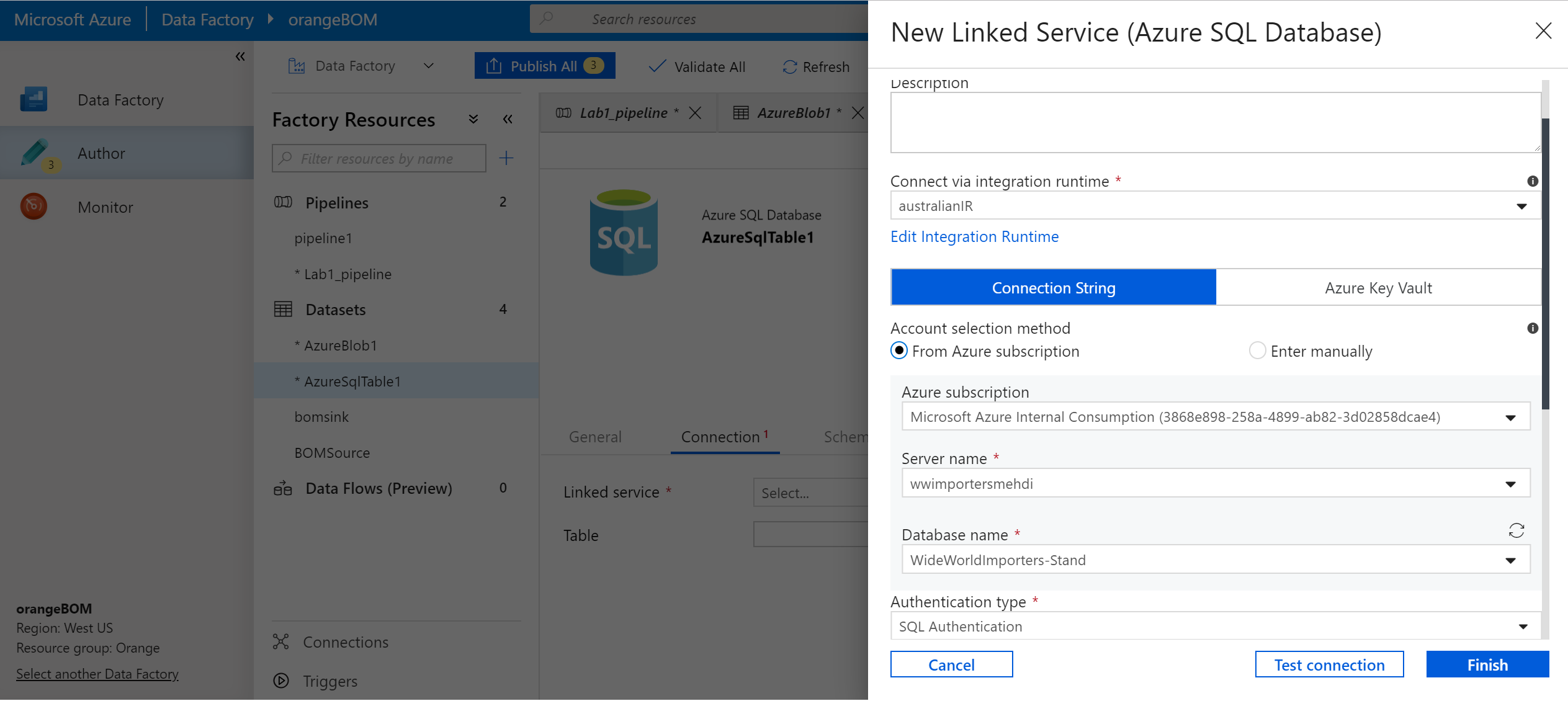
#### Task 3: Create the Sink dataset as an Azure SQL DB

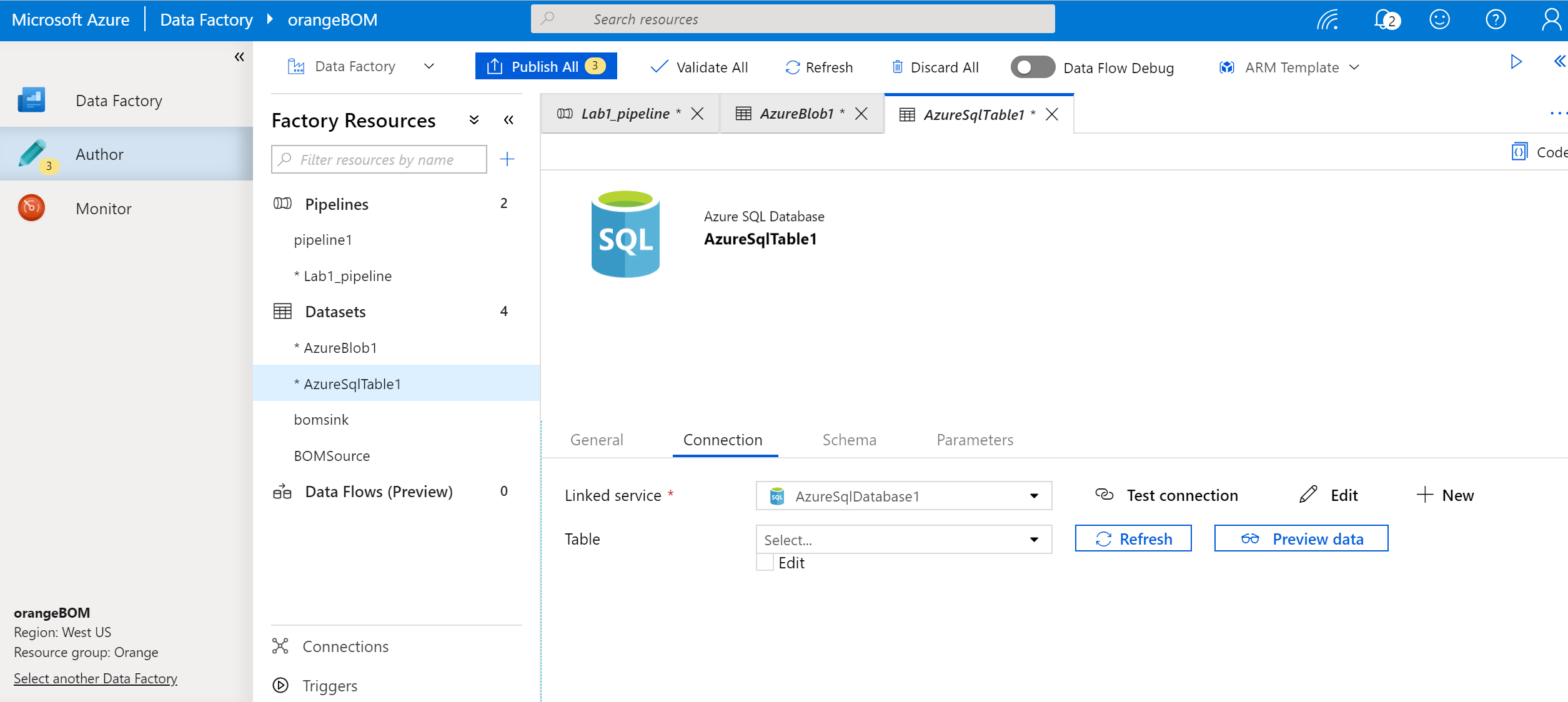
Try setting up the sink data set without going through the steps and practice applying what you learnt in previous step here! (Screen shots of the steps are below, but instructions are intentionally removed for you to only use the screen captures as hints. Ask your instructor if you are stuck in a step and need help.)

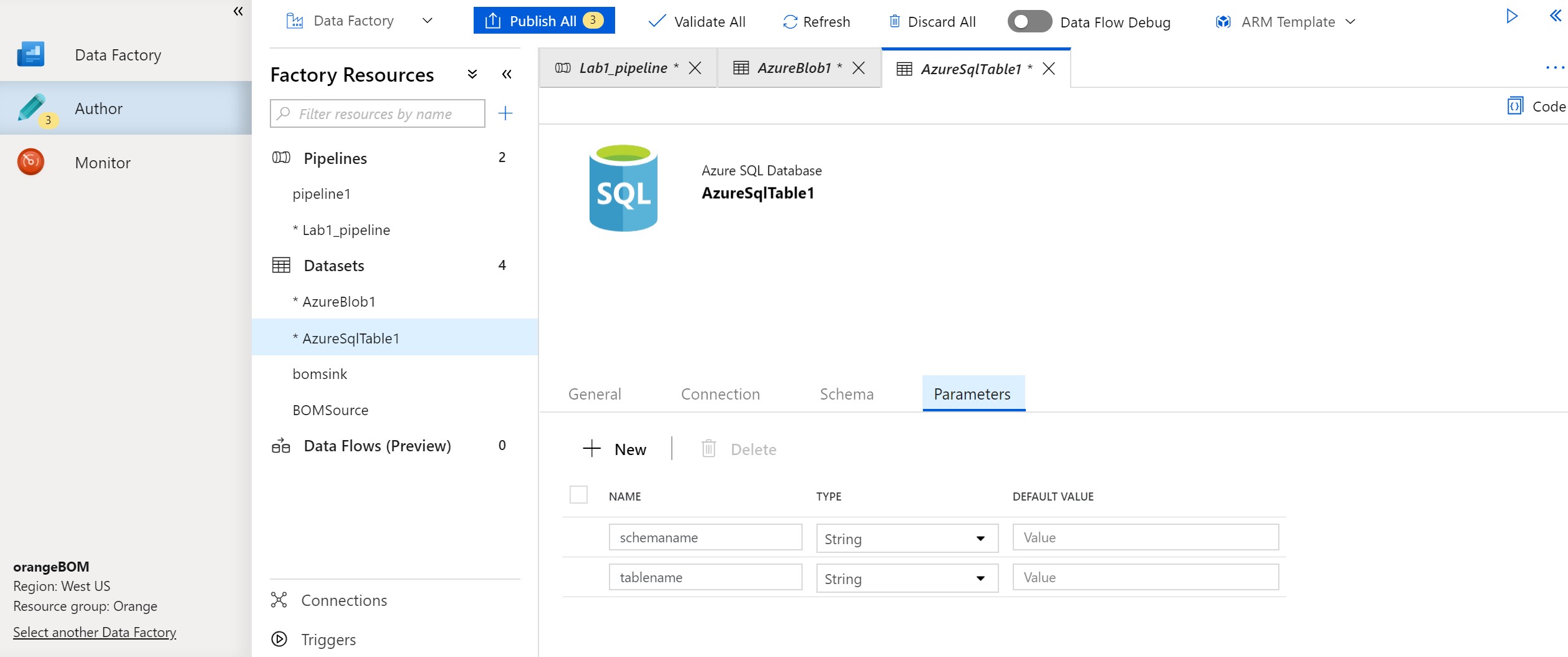
Hint: You need to have your database schema and staging table created before attempting this step! Check the appendix for DDLs



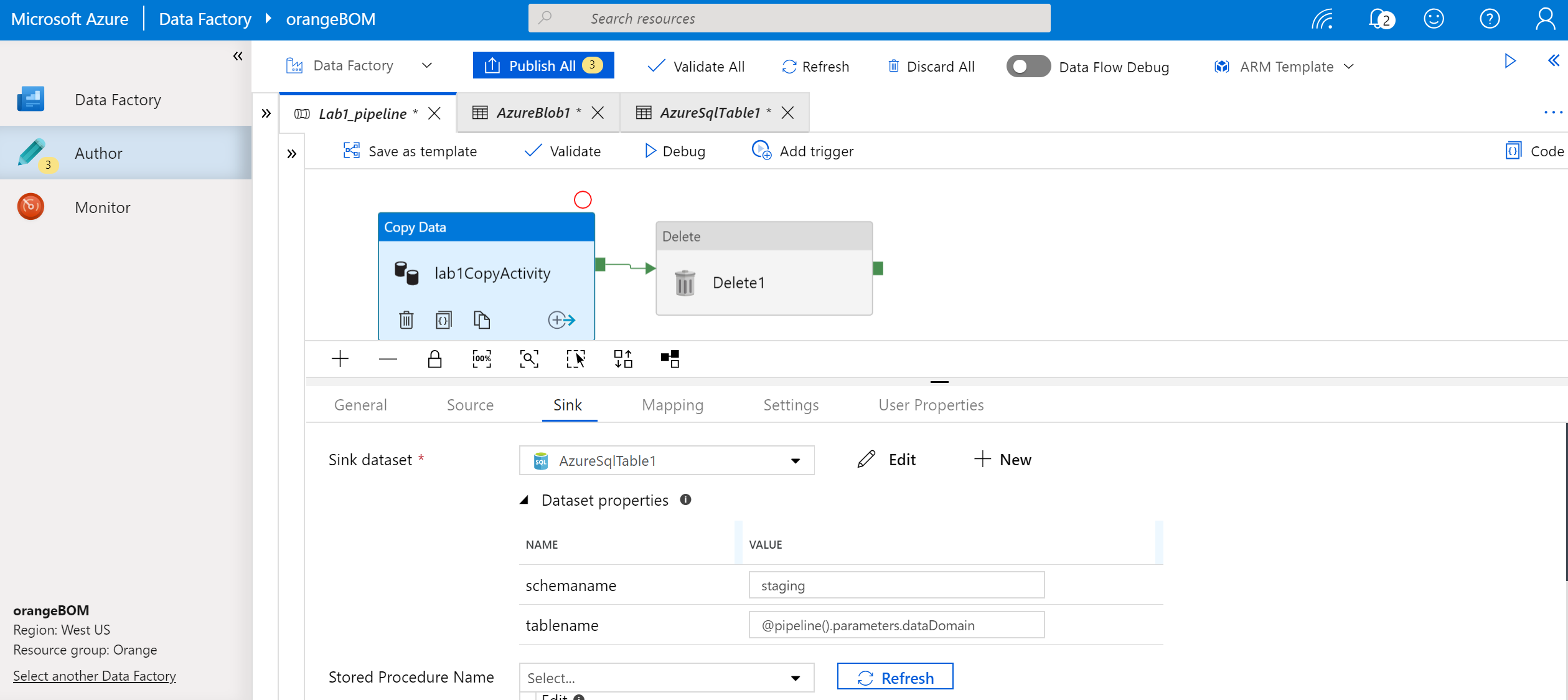


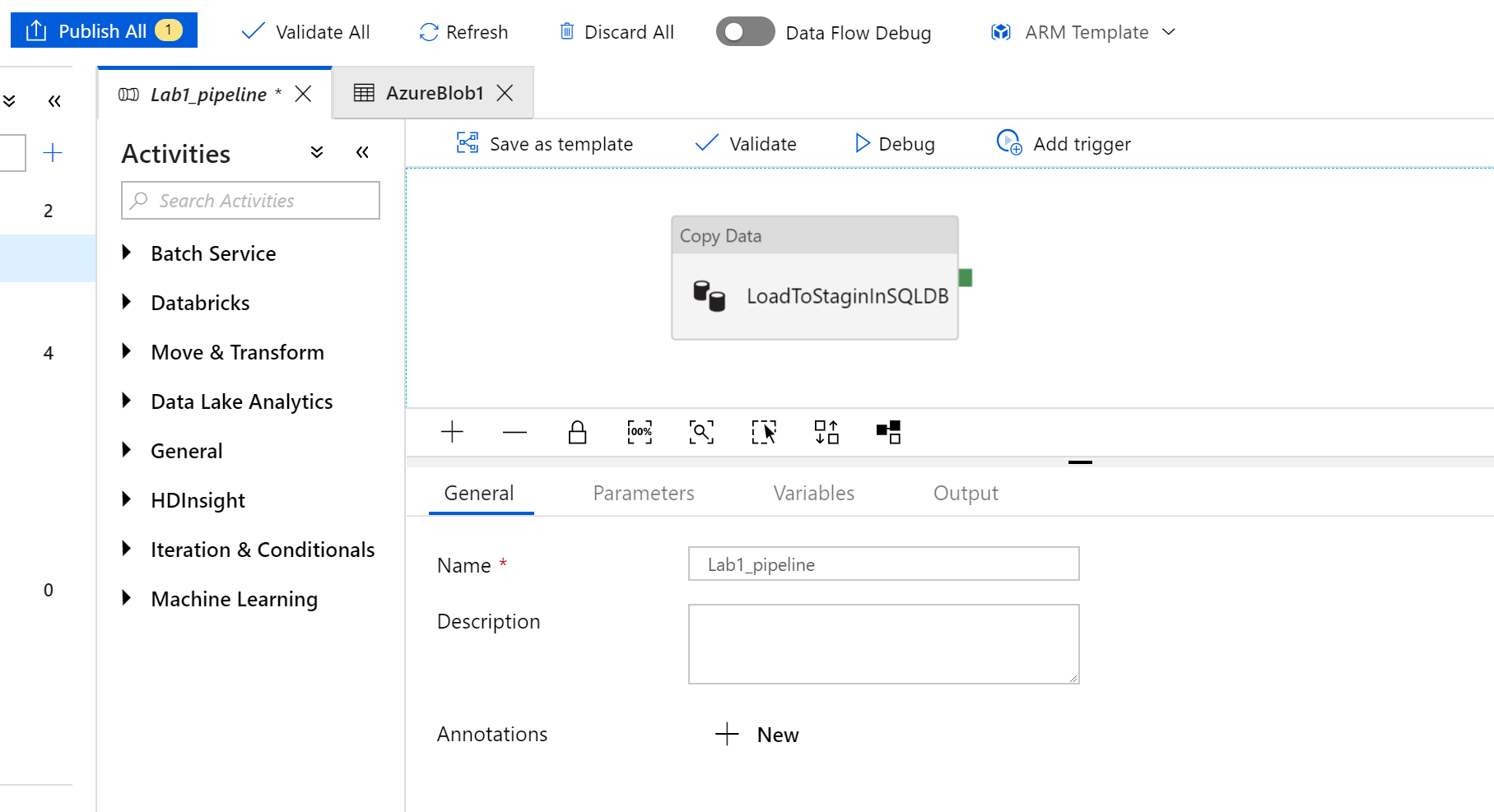










1. Once you setup the source and sink data sets and the pipeline parameters your CopyData Activity is ready!
2. At this stage your pipeline should be like the figure below. Use the “Publish All” button at the top of the page to publis the pipeline. If there is any errors, publish will fail and the errors will be reported to be fixed.****

#### Task 4: Create a stored procedure to perform ETL in SQL DB

1. Create a new schema in your DB and call it “DataWarehouse”
2. Create a new set of tables based on the same DDL but this time add relevant DW columns to it.
3. Create a set stored procs in your SQL DB to perform the ETL in “datawarehouse” schema and name it as <data\_domain>\_elt\_stored\_proc. For example for customer it would be [datawarehouse].[customers\_etl\_stored\_proc]

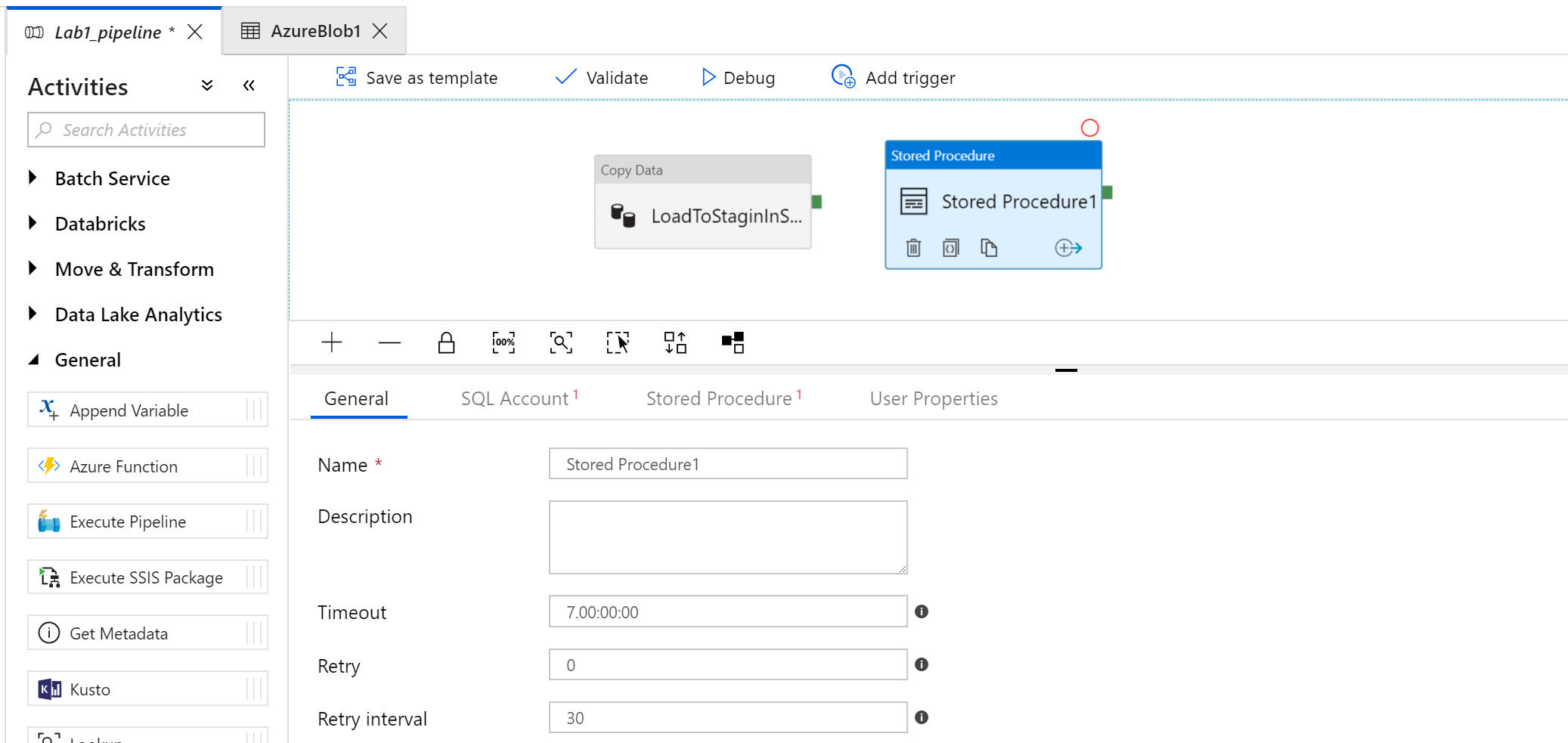
Hint: If it is a fact table and we would not need to keep the history of changes only add a column to store the insert timestamp. If it is a dimension and as per Kimball DW model we are keeping all history (SDC type2) add 5 columns (rec\_start\_dt, rec\_end\_dt, rec\_active\_ind, rec\_insert\_ts, rec\_hash)

There are many ways to perform ETL using SQL for dimensions but generally the steps are:

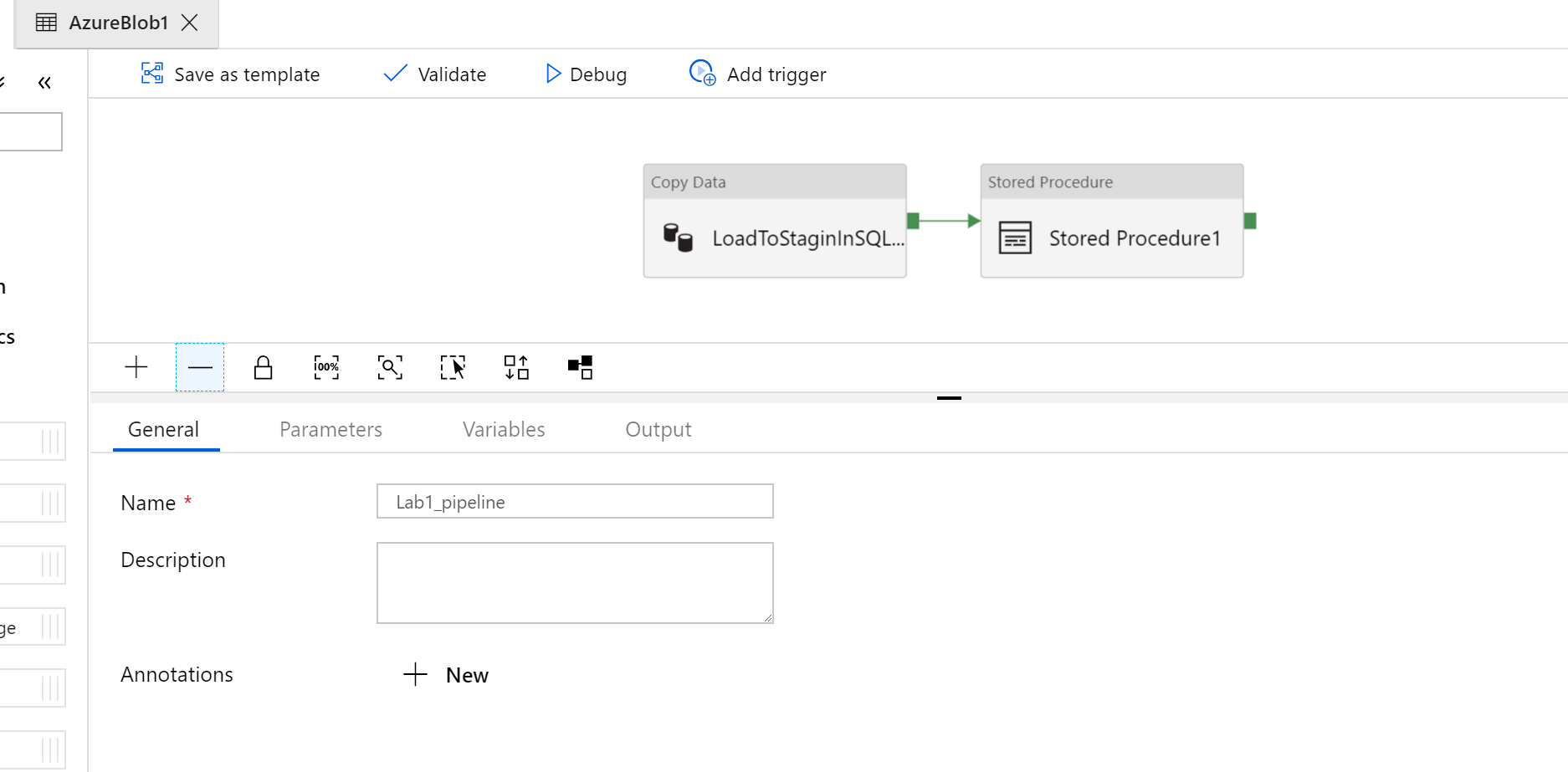
1. Identify the primary key of the table
2. Compare the records in staging with the records currently in DW based on their PK and identify
   1. Records which have never been in the DW table (Insert)
   2. Records which exists in DW at least one of the columns are different (Update + Insert)
   3. Records which exists in the DW and they are identical (Ignore)

#### Task 5: Add a stored procedure activity to the pipeline

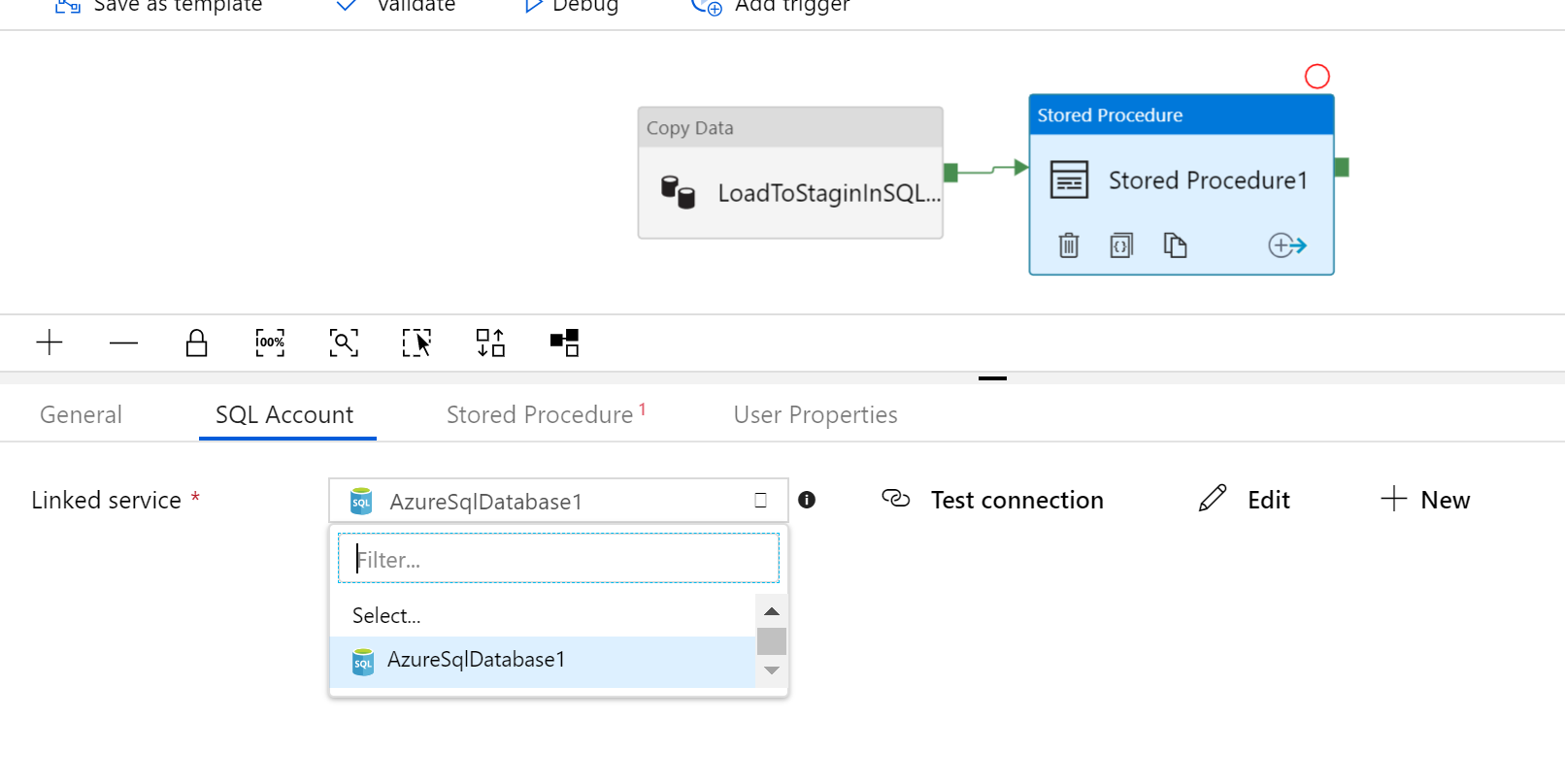
1. Add a stored proc activity to the pipeline.



1. Connect the success from previous activity to the new activity.

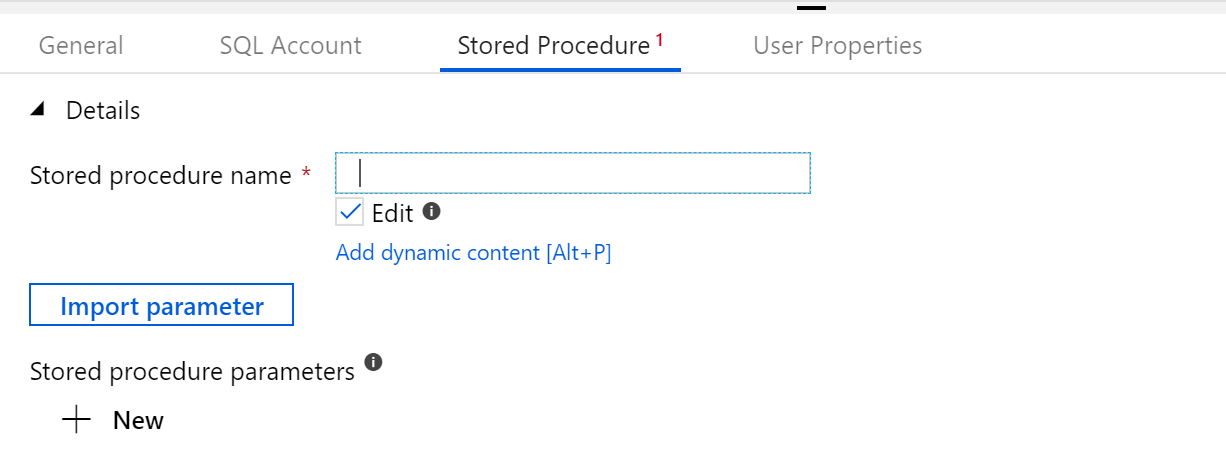


1. Select the SQL Linked Service we created previously in the SQL Account tab.



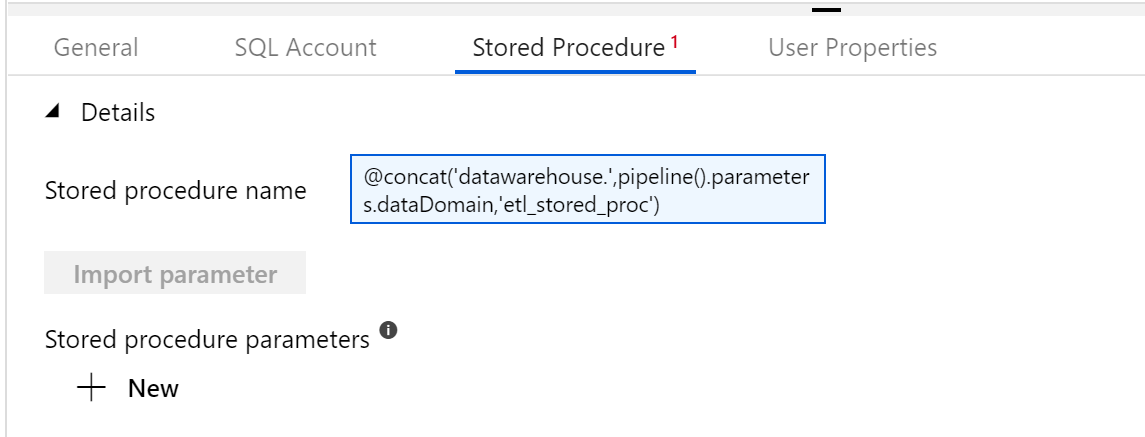
1. In “Stored Procedure” tab check the “Edit” box and click add dynamic content.

Since this pipeline is going to be fully parametrized the stored procedures also will need to be dynamic. We are assuming the name of the stored procs are standardized to <data\_domain>\_etl\_stored\_proc and created in the same “datawarehouse” schema. For example customers ETL stored proc would be [datawarehouse].[customers\_etl\_stored\_proc]



1. In Expression builder enter

@concat('datawarehouse.',pipeline().parameters.dataDomain,'\_etl\_stored\_proc')



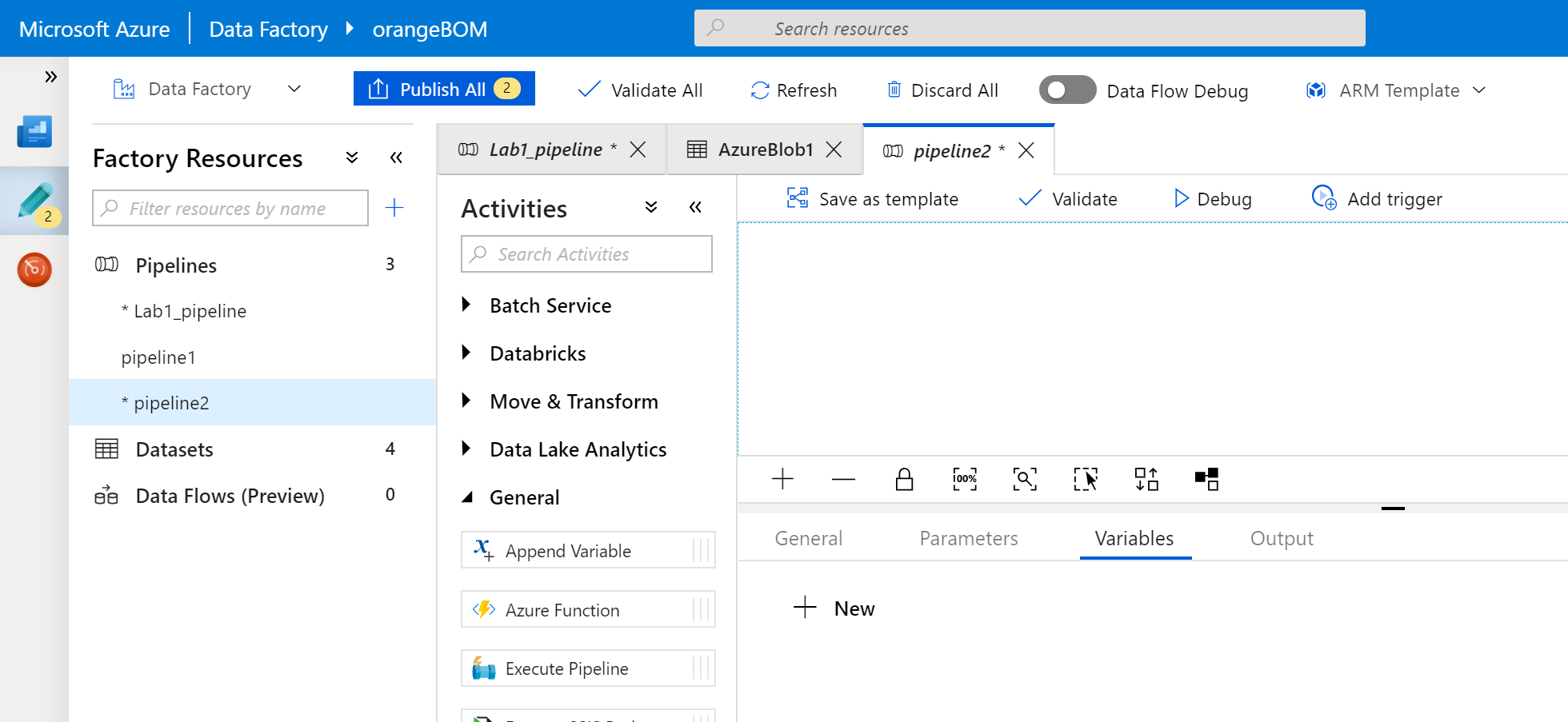
#### Task 6: create a pipeline to run daily load

Now that we create a parametrized pipeline, we need to create other pipelines to generate the date parameter and run this pipeline.

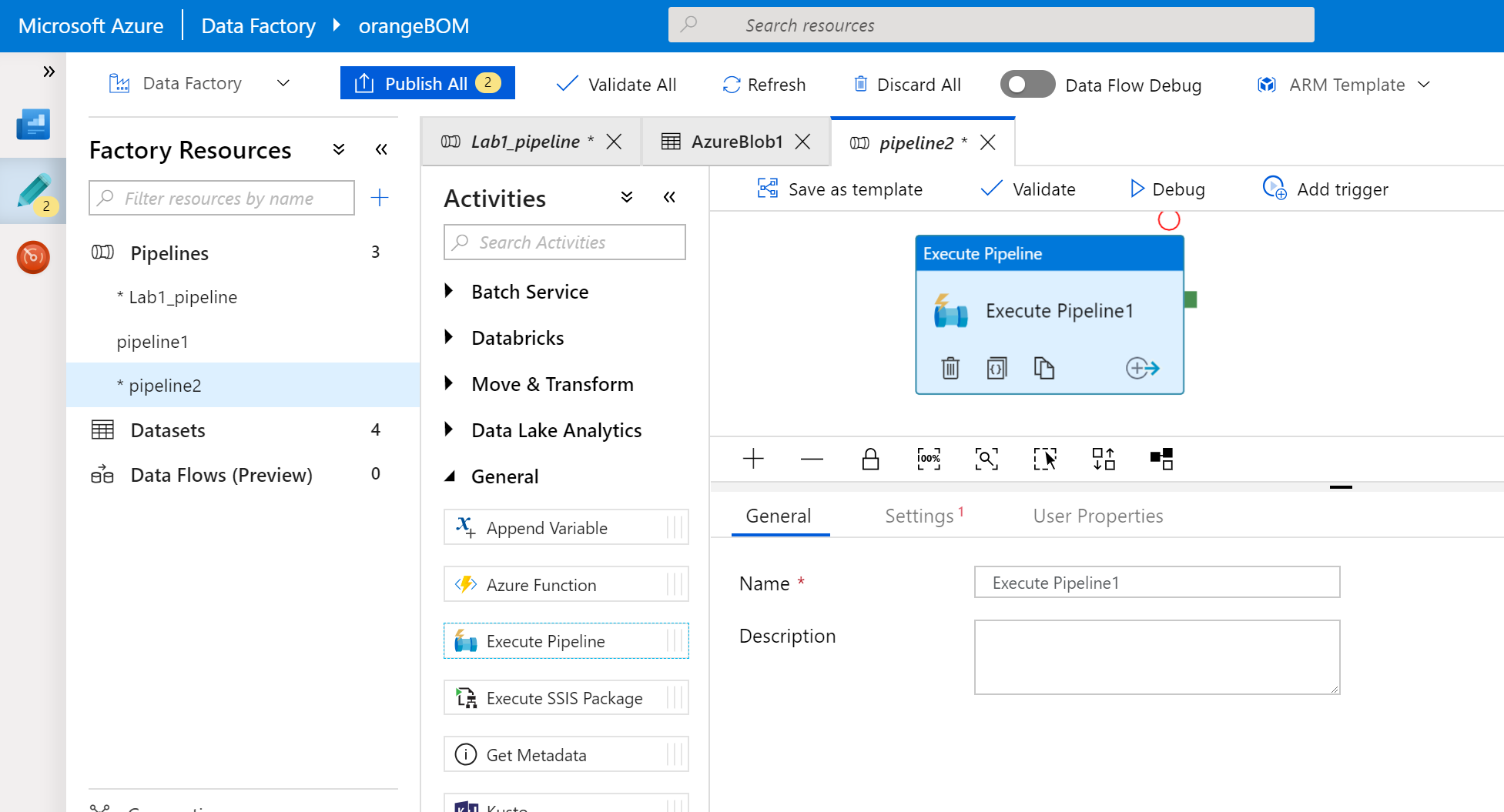
The primary advantage of the pipeline we just built is that

* + 1. It can be used for any table if the tables follow the same file and table naming standards.
    2. It can be used to perform ETL for both daily loads and historical iterative loads.

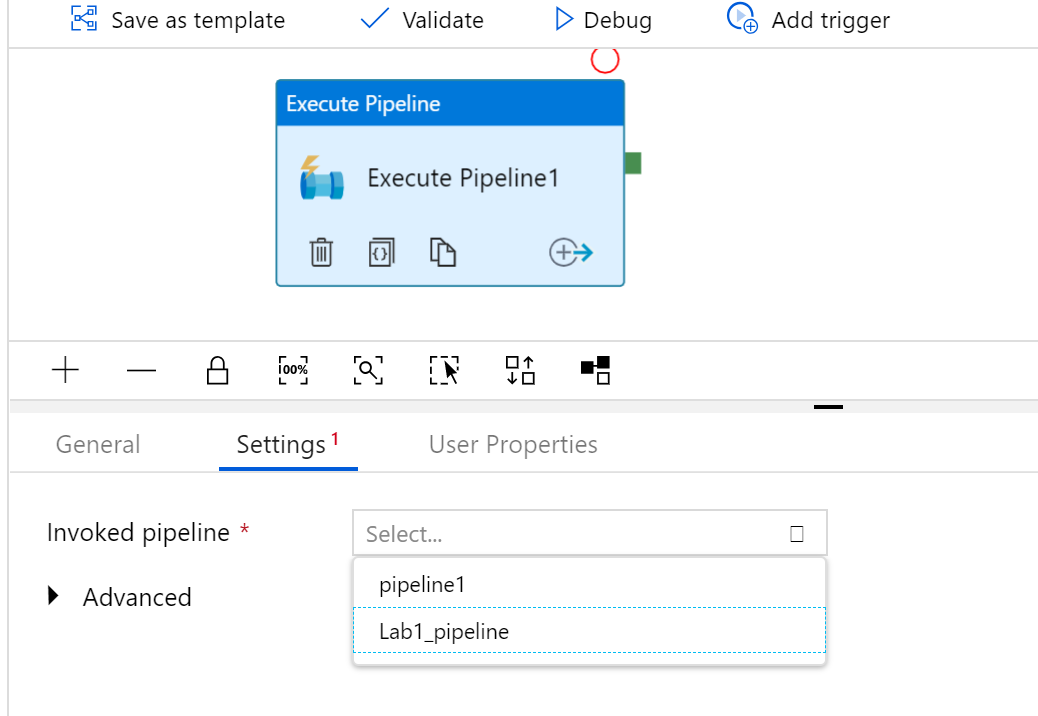
1. Create a new pipeline



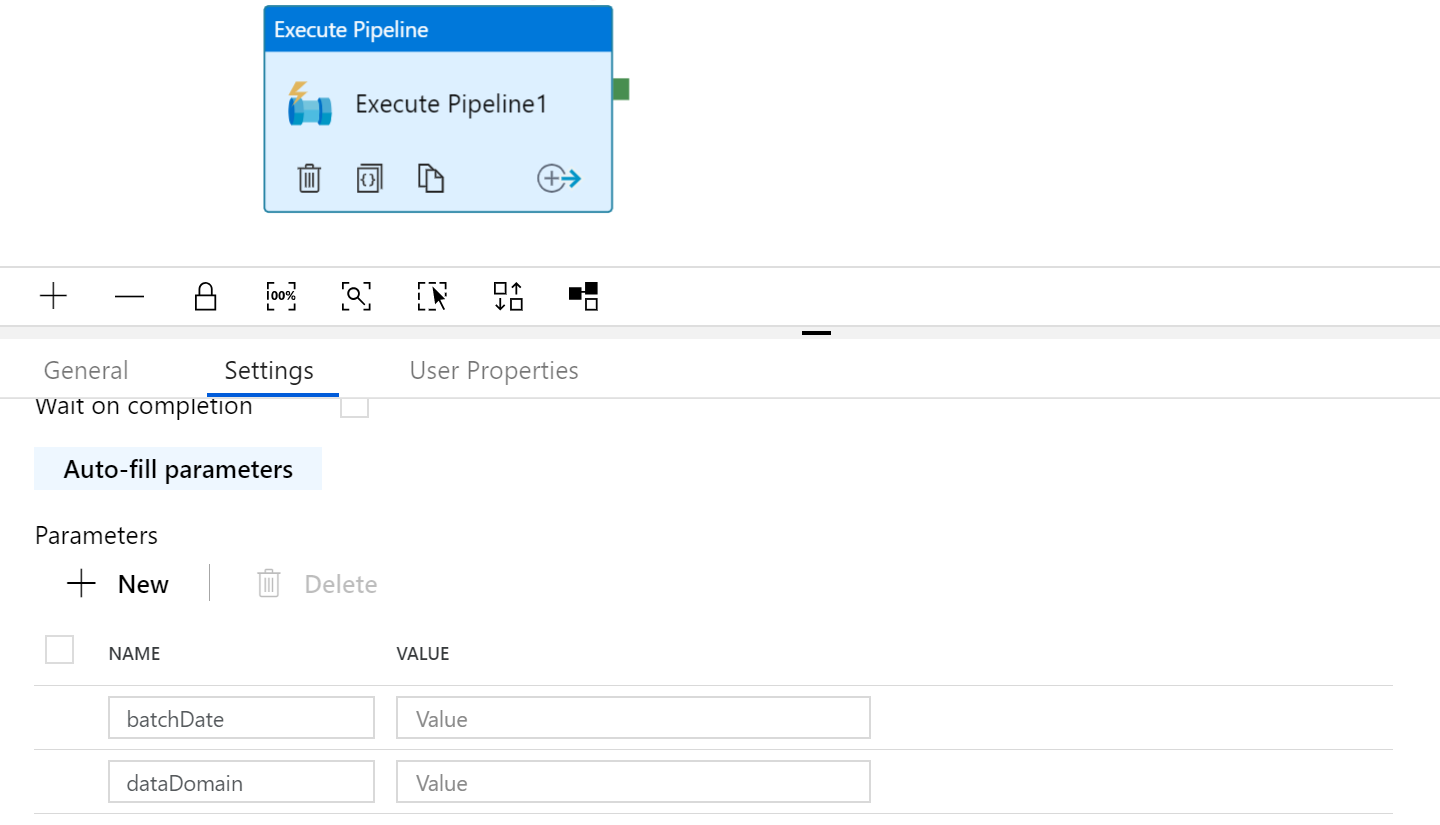
1. Add “Execute Pipeline” activity to the new pipeline



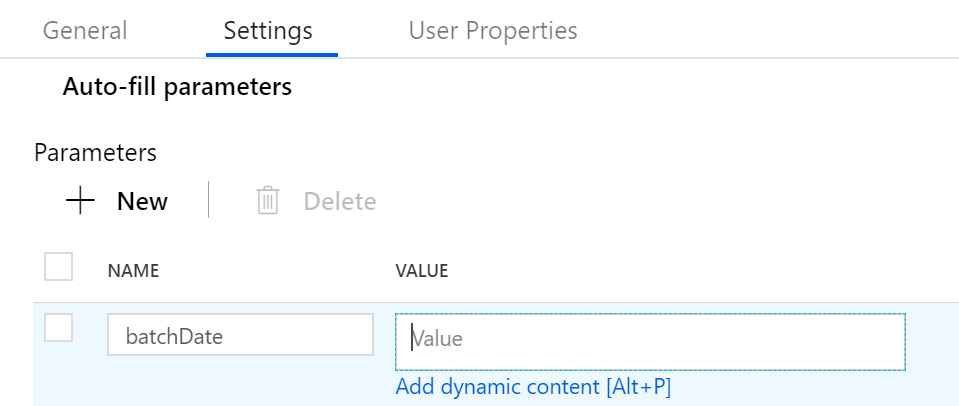
1. Rename it to something meaning full (e.g. customer\_daily\_load or customer\_historical\_load)
2. In Settings tab select the pipeline created in previous Task



1. Extend the “Advanced” part and click “Auto Fill Parameters” to add the inner pipeline prams to

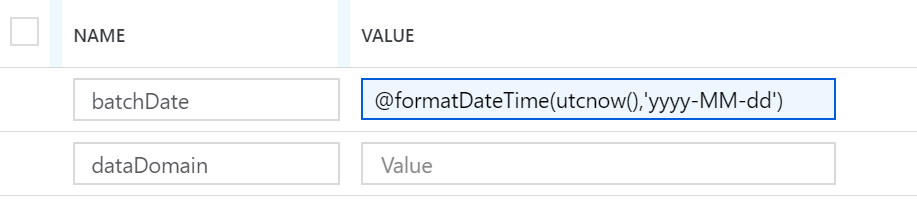


1. For “batchDate” value click “Add Dynamic Content” and open the expression builder

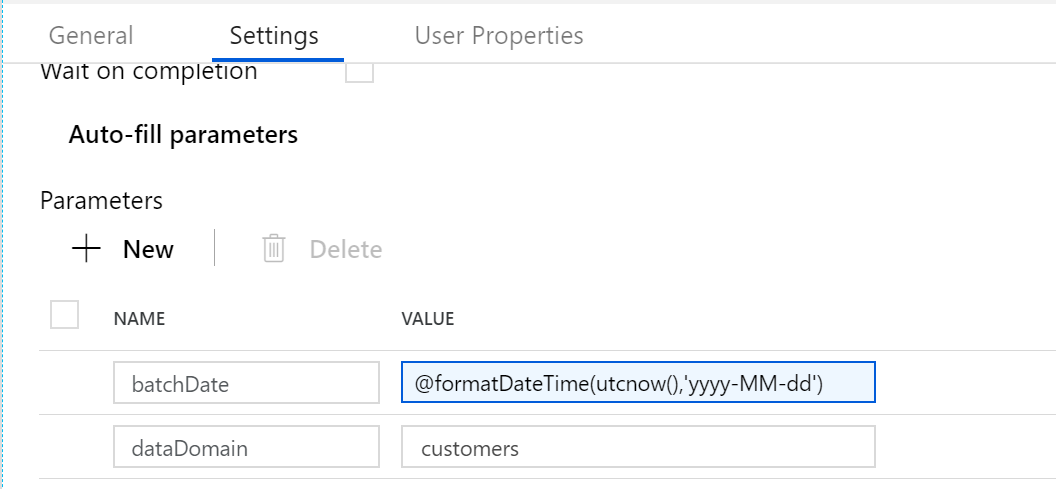


1. Paste the below expression code. This expression code generates todays datetime (in UTC time) and the outer function format it to ‘yyyy-MM-dd’ format. (This is the format of date in our filenames)

@formatDateTime(utcnow(),'yyyy-MM-dd')

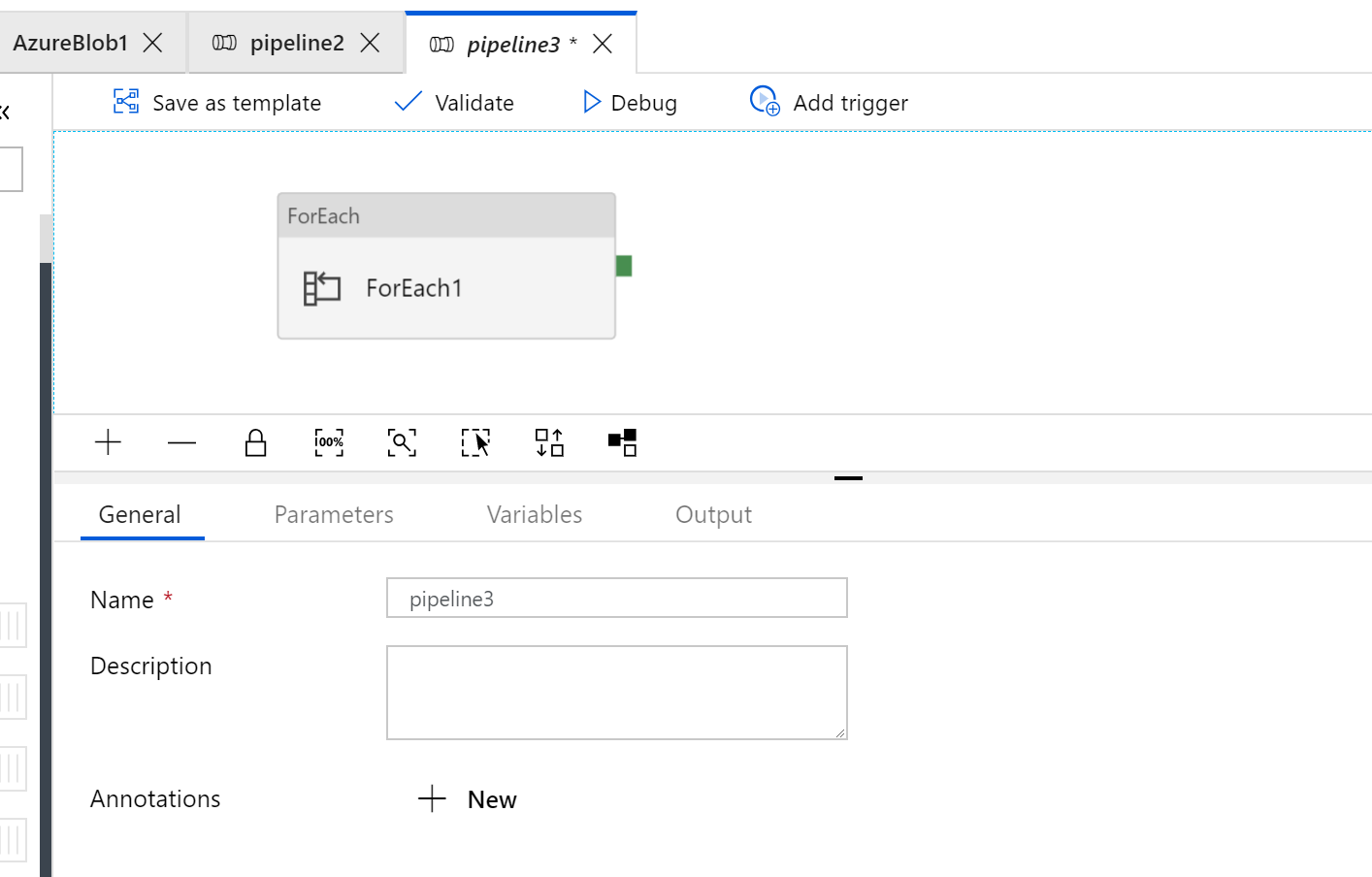


1. For dataDomain we can enter either “customers” or “orders” or “orderlines”



#### Task 7: create a pipeline to iteratively perform historical load

1. Create a new pipeline and name it for example customers\_historical\_load
2. Drag a “ForEach” activity

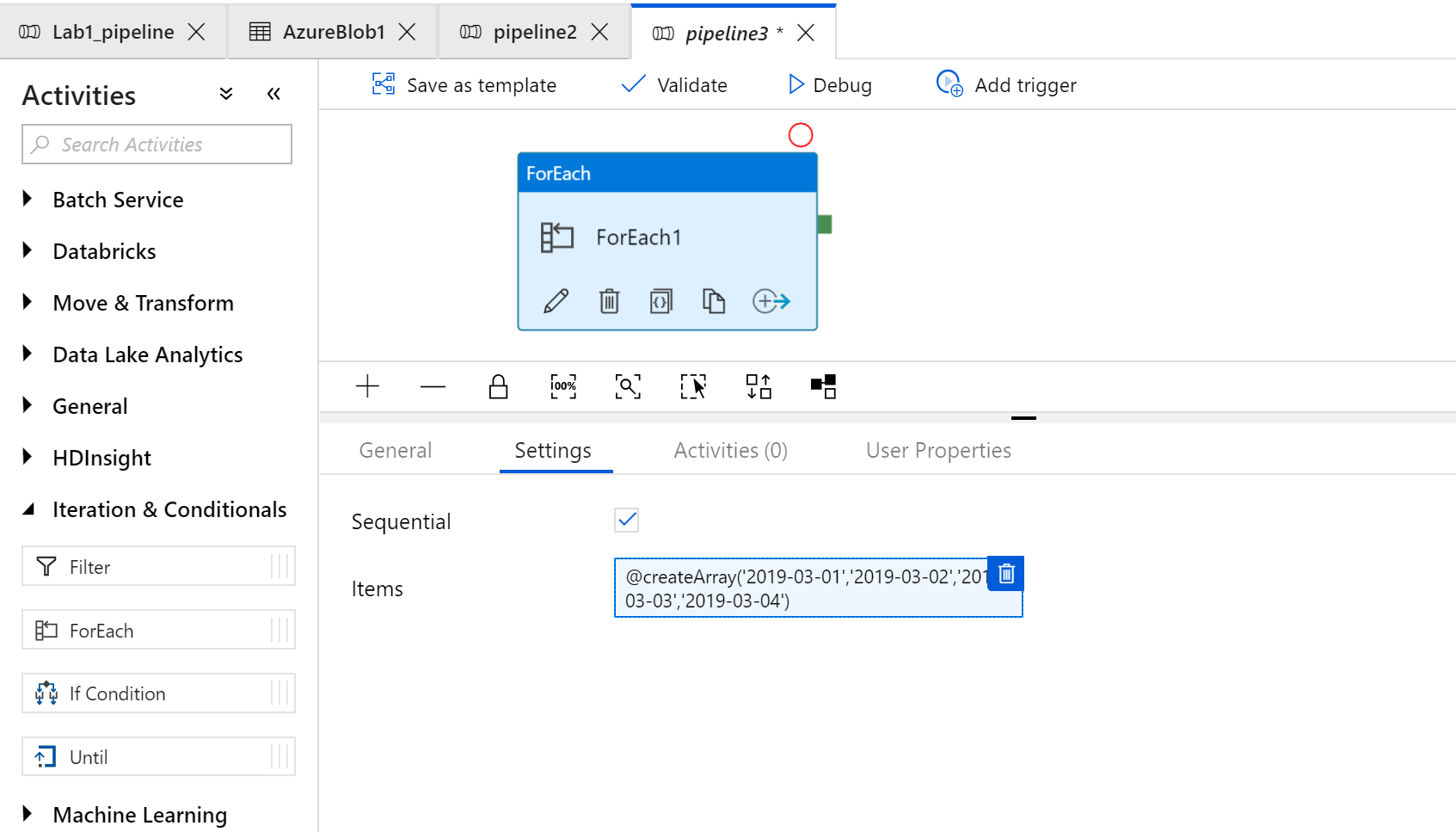


1. Under settings tab select “Sequential” (Can you guess why this option is being selected?)
2. for items open the expression builder and paste the following code:

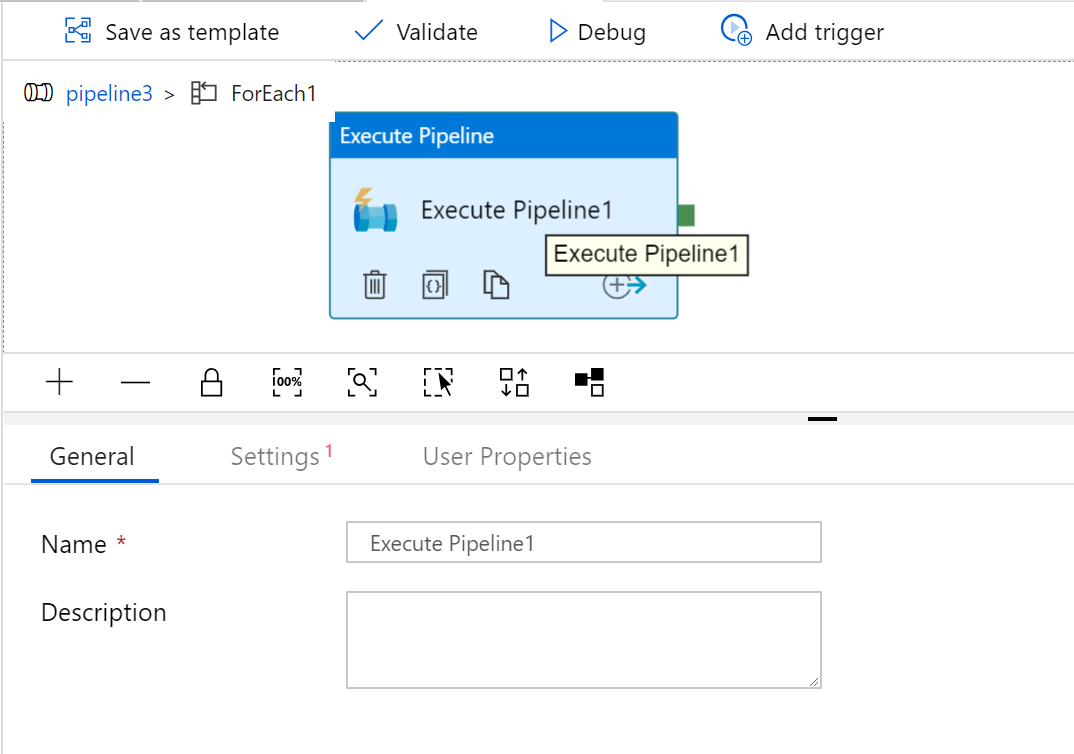
@createArray('2019-02-07','2019-02-08','2019-02-09','2019-02-10')

**Note:** The “createArray()” expression function converts the string items we provide to an array which is what the “Foreach” activity expects.

**Challenge:** Can you modify the expression to accept a start and an end date and generate all the dates for historical load. For example, if we are building a dimension table for source files between 2019-01-01 to 2019-04-01.

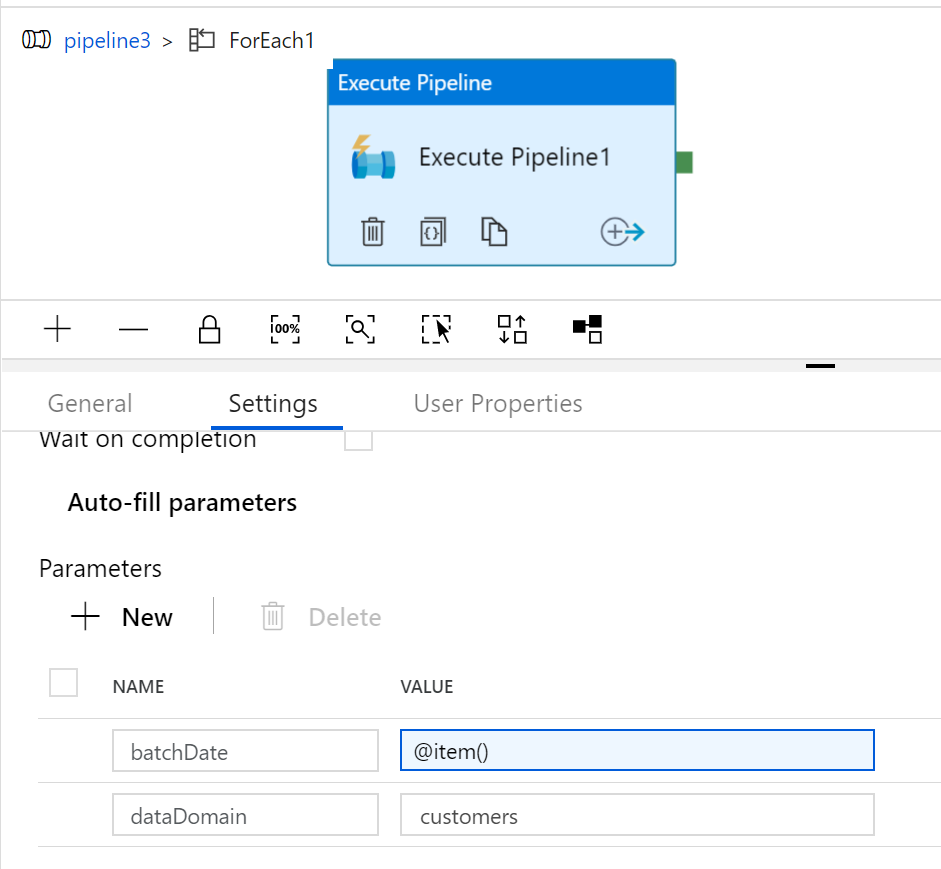


1. Double click on the “Foreach” activity to open it.
2. Drag an “Execute Pipeline” activity in the new canvas.



1. In Settings tab select the pipeline created in previous Task
2. For “batchDate” value click “Add Dynamic Content” and open the expression builder and enter

@item()

1. For dataDomain parameter value enter either “customers”, “orders” or “orderlines”
2. Publish your pipeline.

## After the hands-on lab

Duration: 10 minutes

In this exercise, attendees will deprovision any Azure resources that were created in support of the lab.

#### Task 1: Delete resource group

1. Using the Azure portal, navigate to the Resource group you used throughout this hands-on lab by selecting **Resource groups** in the left menu.
2. Search for the name of your research group and select it from the list.
3. Select **Delete** in the command bar and confirm the deletion by re-typing the Resource group name and selecting **Delete**.

You should follow all steps provided *after* attending the Hands-on lab.

## Appendix

#### Customer Table DDL:

CREATE TABLE [Datawarehouse].[Customers](

    [CustomerID] [int] NOT NULL,

    [CustomerName] [nvarchar](100) NOT NULL,

    [BillToCustomerID] [int] NOT NULL,

    [CustomerCategoryID] [int] NOT NULL,

    [BuyingGroupID] [int] NULL,

    [PrimaryContactPersonID] [int] NOT NULL,

    [AlternateContactPersonID] [int] NULL,

    [DeliveryMethodID] [int] NOT NULL,

    [DeliveryCityID] [int] NOT NULL,

    [PostalCityID] [int] NOT NULL,

    [CreditLimit] [decimal](18, 2) NULL,

    [AccountOpenedDate] [date] NOT NULL,

    [StandardDiscountPercentage] [decimal](18, 3) NOT NULL,

    [IsStatementSent] [bit] NOT NULL,

    [IsOnCreditHold] [bit] NOT NULL,

    [PaymentDays] [int] NOT NULL,

    [PhoneNumber] [nvarchar](20) NOT NULL,

    [FaxNumber] [nvarchar](20) NOT NULL,

    [DeliveryRun] [nvarchar](5) NULL,

    [RunPosition] [nvarchar](5) NULL,

    [WebsiteURL] [nvarchar](256) NOT NULL,

    [DeliveryAddressLine1] [nvarchar](60) NOT NULL,

    [DeliveryAddressLine2] [nvarchar](60) NULL,

    [DeliveryPostalCode] [nvarchar](10) NOT NULL,

    [DeliveryLocation] [geography] NULL,

    [PostalAddressLine1] [nvarchar](60) NOT NULL,

    [PostalAddressLine2] [nvarchar](60) NULL,

    [PostalPostalCode] [nvarchar](10) NOT NULL,

    [LastEditedBy] [int] NOT NULL,

    [ValidFrom] [datetime2](7) NOT NULL,

    [ValidTo] [datetime2](7) NOT NULL

)

GO

#### Orders Table DDL:

CREATE TABLE [datawarehouse].[Orders](

    [OrderID] [int] NOT NULL,

    [CustomerID] [int] NOT NULL,

    [SalespersonPersonID] [int] NOT NULL,

    [PickedByPersonID] [int] NULL,

    [ContactPersonID] [int] NOT NULL,

    [BackorderOrderID] [int] NULL,

    [OrderDate] [date] NOT NULL,

    [ExpectedDeliveryDate] [date] NOT NULL,

    [CustomerPurchaseOrderNumber] [nvarchar](20) NULL,

    [IsUndersupplyBackordered] [bit] NOT NULL,

    [Comments] [nvarchar](max) NULL,

    [DeliveryInstructions] [nvarchar](max) NULL,

    [InternalComments] [nvarchar](max) NULL,

    [PickingCompletedWhen] [datetime2](7) NULL,

    [LastEditedBy] [int] NOT NULL,

    [LastEditedWhen] [datetime2](7) NOT NULL

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

#### orderlines Table DDL:

CREATE TABLE [datawarehouse].[OrderLines](

    [OrderLineID] [int] NOT NULL,

    [OrderID] [int] NOT NULL,

    [StockItemID] [int] NOT NULL,

    [Description] [nvarchar](100) NOT NULL,

    [PackageTypeID] [int] NOT NULL,

    [Quantity] [int] NOT NULL,

    [UnitPrice] [decimal](18, 2) NULL,

    [TaxRate] [decimal](18, 3) NOT NULL,

    [PickedQuantity] [int] NOT NULL,

    [PickingCompletedWhen] [datetime2](7) NULL,

    [LastEditedBy] [int] NOT NULL,

    [LastEditedWhen] [datetime2](7) NOT NULL

) ON [PRIMARY]