Basic ETL Processing with Azure Data Factory - 2/4

qimia.io/en/blog/Basic-ETL-Processing-with-Azure-Data-Factory

Welcome back to the second article of our Azure ETL series. In the last article, we saw how to create resources in Azure.

In this episode, we will show you how to set up an Azure Data Factory (ADF), how to handle costs within ADF, and how to process a sample dataset. As described in the introduction, we use the Northwind data, load it to an MS SQL database and dump it from there to Azure Data Lake storage in a daily running procedure using ADF.

1. Azure Data Factory

Azure Data Factory is a cloud-based ETL and data integration service to create workflows for moving and transforming data. With Data Factory you can create scheduled workflows (pipelines) in a code-free manner. Each graphically programmed block in the pipeline represents a piece of JSON-code that can also be coded manually or ingested from a GIT repository instead of dragging and dropping. The basic workflow in Data Factory is structured in pipelines. The data in- and outputs (here sources and sinks) are defined as so called datasets. Each Data Factory usually consists of one or more pipelines that contain and execute activities.

The concepts of datasets, pipelines, and activities are explained later in this article. Any changes to a Data Factory have to be published in order to be saved for the next session unless you have enabled version control! Also, keep in mind that nearly everything that you do in ADF has a price tag:

Not only the execution of a workflow but also building a pipeline, monitoring runs, or just storing the code. However, most of it is quite cheap, such that mainly processing time matters:

- Runtime costs are $0.25/DIU-hour[1] for data movement and $0.005/hour per activity
- Runtime of a data flow costs $0.193-$0.343 per vCore-hour depending on the types of cores
- Each piece of runtime is calculated per single activity and rounded up to a minute! => It is more cost-effective to have few complex activities with long lasting execution than many simple ones.

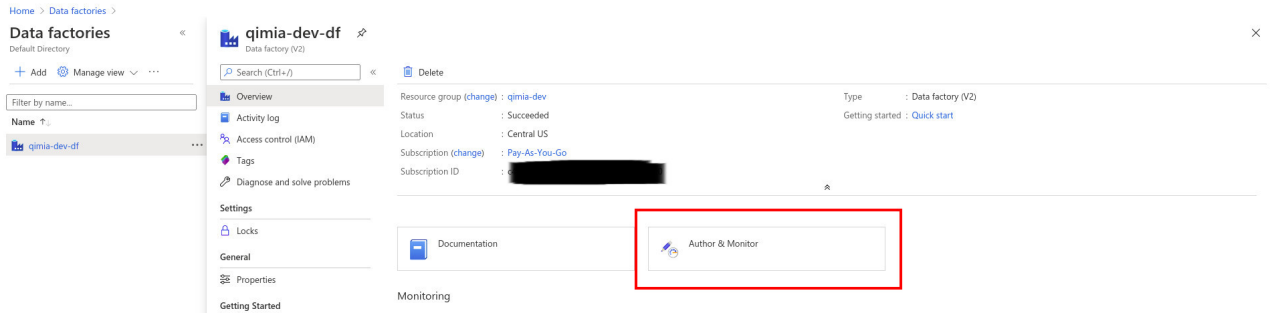
[1] DIU-hour = Data Integration Unit hour, essentially compute nodes per hour

1.1. Creating an Azure Data Factory

In order to create a Data Factory resource, go to the Azure Portal. Search for „data factories“ and create/add one. Choose a globally unique name. Choose the **Version 2**.

Place the Data Factory in the resource group you created and choose the same region (=location) as for the storage and resource group that you created in the first episode. You don't have to enable GIT for now. However, it is a best practice to include some version control. You can also use the Azure DevOps repo for it.

Every publishing you do on ADF will also be pushed to that repo. However, saving your code in the repo does not automatically publish it. We provide the complete code for our ADF as JSONs on our Github. You can add the JSONs to your repo and deploy a Data Factory including the resource from the repo. Now, everything is in place for getting started with the ETL preparation. You can go to your Data Factory and enter it by clicking "Author & Monitor".



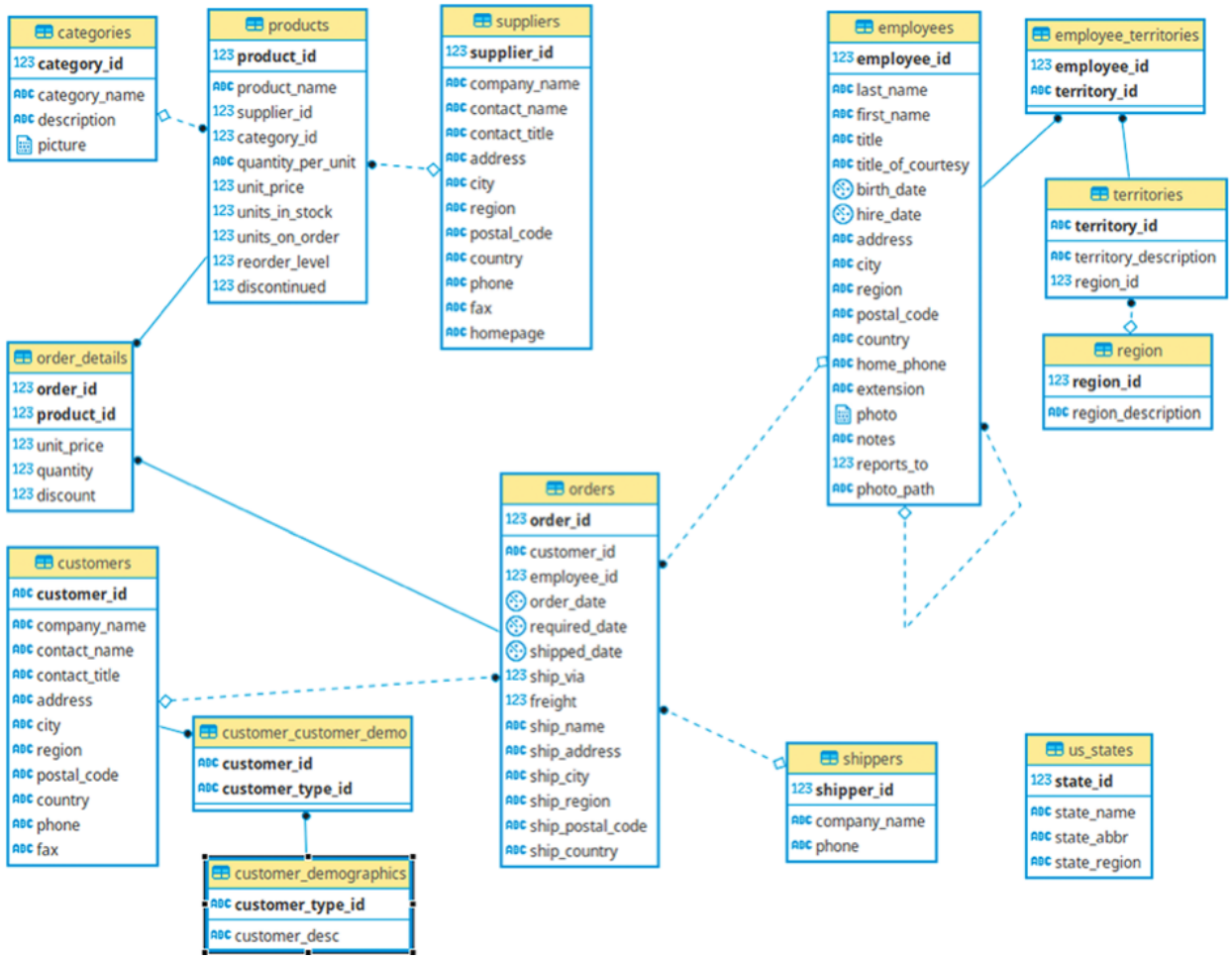
Creating an Azure Data Factory

2. Basic ETL Example

In this section, we will create an easy pipeline that already contains a lot of tools in ADF. Furthermore, we explain the basic ideas of it, such as linked services, datasets, activities, and pipelines.

2.0. Preparation

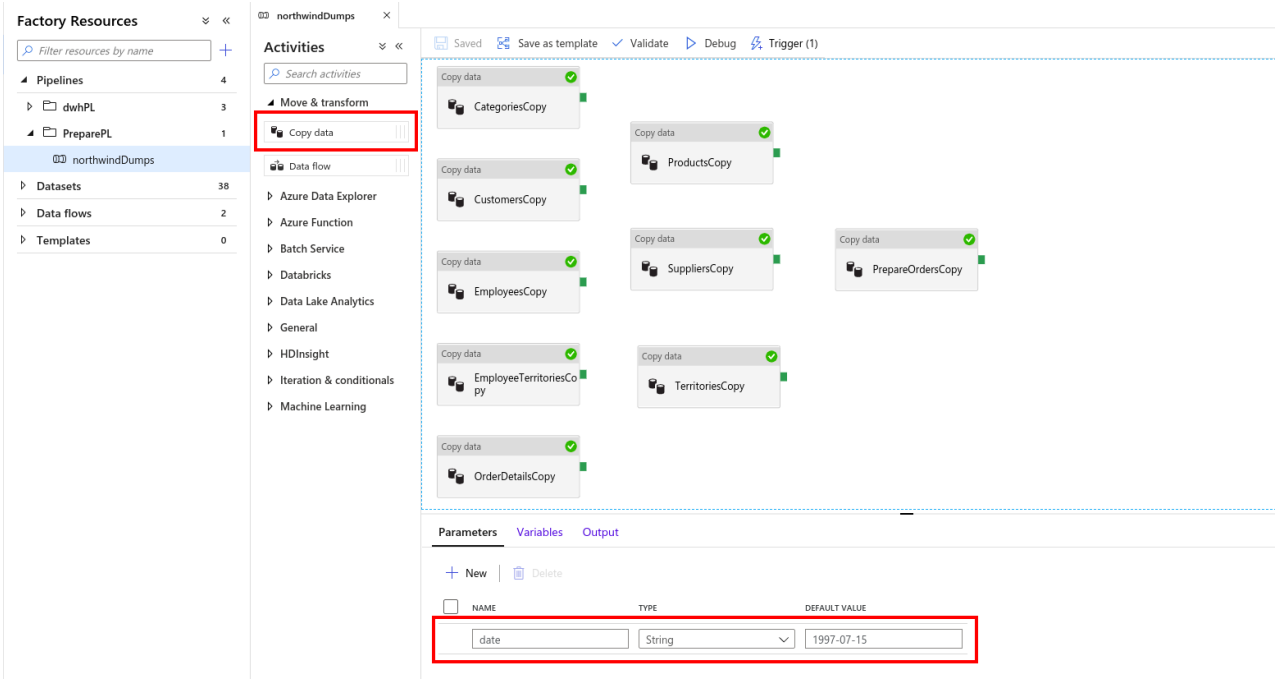
As mentioned before, we will inject the data into the Azure SQL-Server. The data itself can be found here as an SQL file containing CREATE and INSERT statements. Creating and entering an Azure SQL DB is quite similar to creating the Synapse DWH, which will be done in part 3. If you are not sure about it, you can check it out in our next article Creating the Data Warehouse Schema. The Northwind data is organized in an OLTP-like data schema:



Northwind Data Schema
Northwind Data Schema Image Source

2.1. The Pipeline

In Data Factory a pipeline is a group of activities, each of them performing pieces of the workflow such as copy, transform or verify data. These activities are brought together in a DAG-like graphical programming interface. In order to control the workflow, a pipeline has two other basic features: Triggers and Parameters/Variables. Furthermore, the pipeline can change the workflow, if a failure occurs. In our scenario, we just create one pipeline.



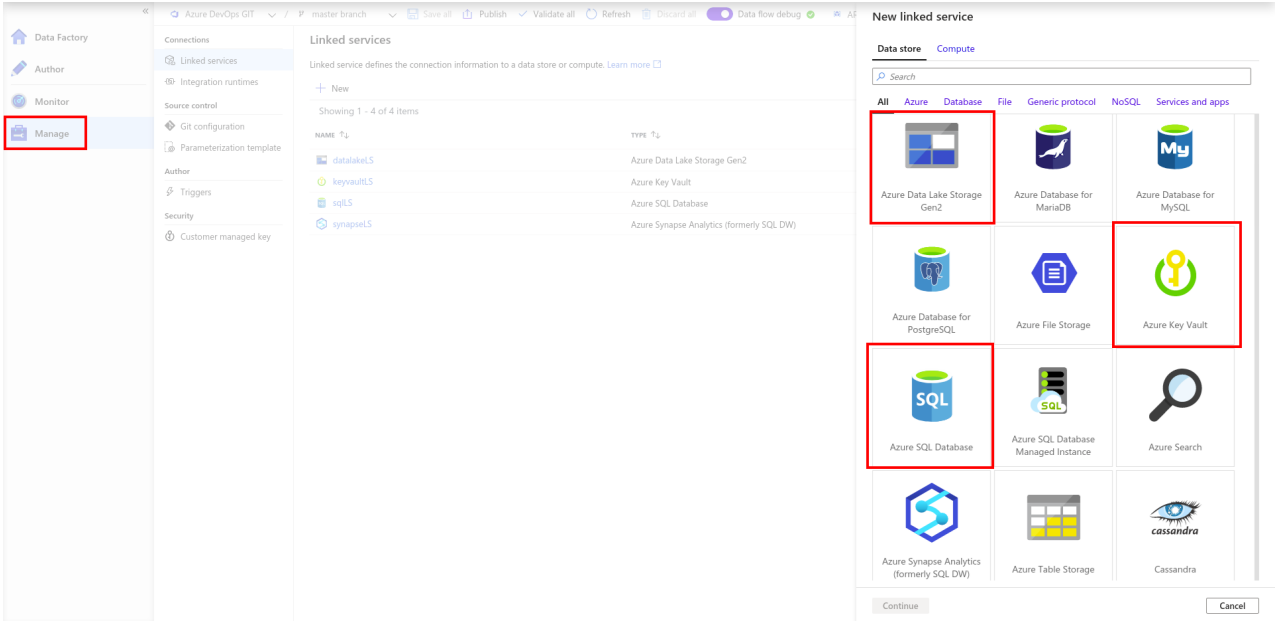
Basic ETL Example - The Pipeline
The copy activities in the preparation pipeline do not have any dependencies. They are all executed in parallel. Here we just put them next to each other for a better overview.

It reads partitions from the orders table together with all other needed tables and dumps them to ADLS. This pipeline just contains nine copy-activities. One for each table that we use for the DWH model: Categories, Customers, Employees, Employee-Territories, Order Details, Products, Suppliers, Territories, and Orders. Furthermore, it includes one Parameter "date". This parameter can later be used for scheduling. Each copy activity has a Source and a Sink. For both, you need to create a dataset, which is tied to linked services.

2.1.1. Linked Services and Datasets

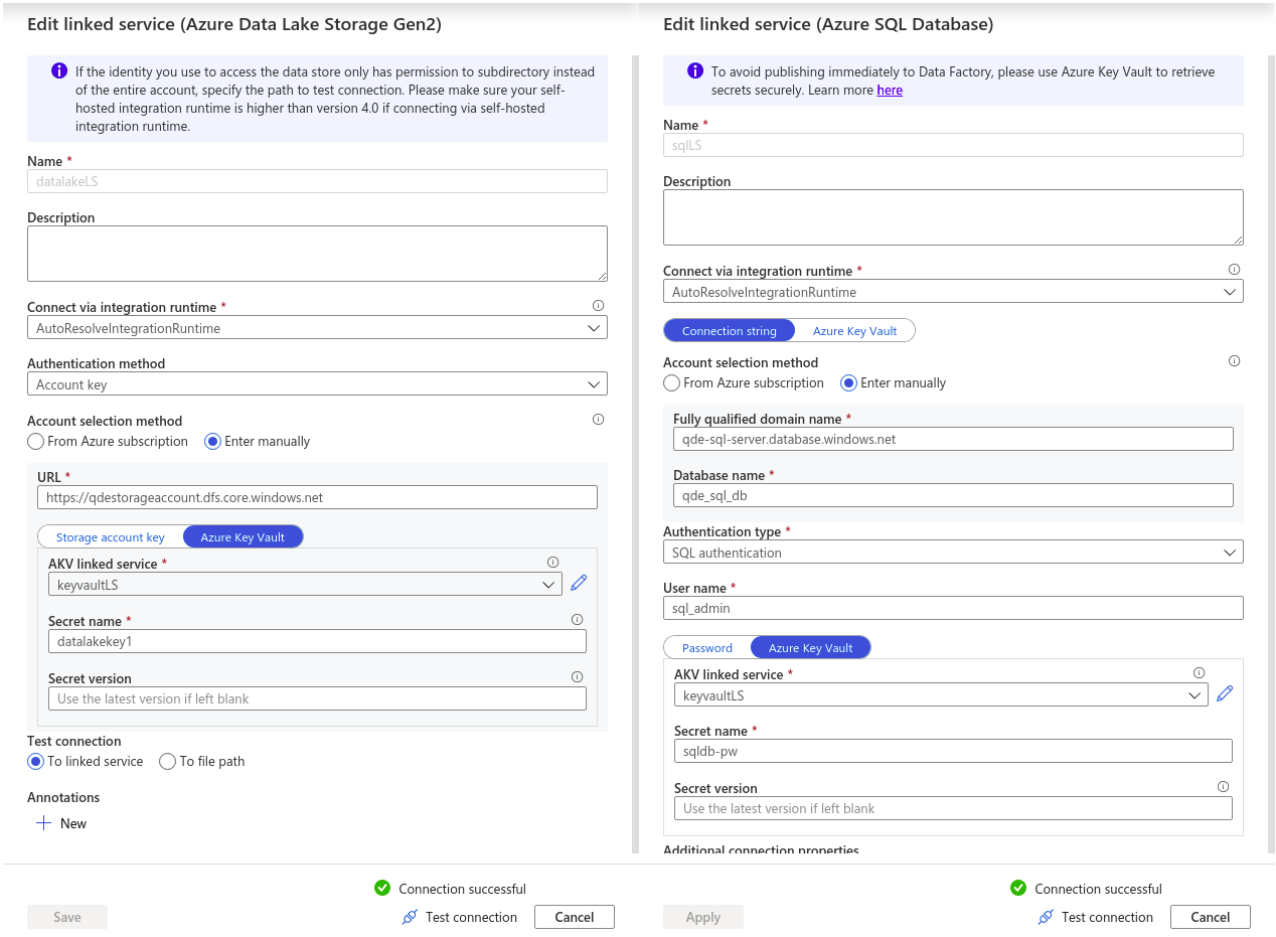
The Linked Service (LS) is essentially a connection string to the data that has to be processed together with the runtime that should be used for it. A dataset is basically a view or a representation of data, which is used by activity within a pipeline. In order to create an LS, you need to switch to the "Manage"-tab in your ADF. Here we need three linked services:

One for the Key Vault, one for the storage, and one for the SQL database. Go to the Manage-tab and create the linked services. Choose the according tiles. In this example, we provide the access key to the storage via Key Vault. This is the more secure way as is suggested by Azure.

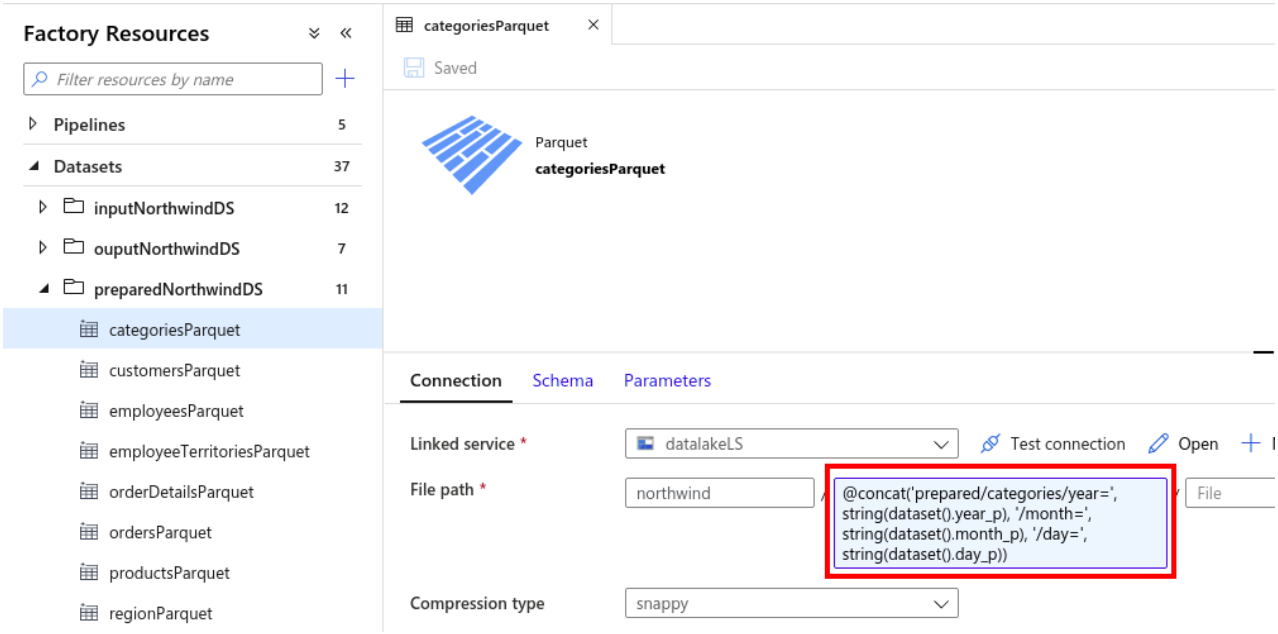


Azure Data Factory - The Pipeline - Linked Services and Datasets I
Create the Key Vault linked service first. You will be asked to grant Data Factory service access to the Key Vault. Copy the object ID and click that link. You will be redirected to a page in the Key Vault, where you can add access policies.

Add an access policy by giving it the needed secret permissions and your object ID. After adding the policy, make sure to hit "save". For the other Services, you can use the key vault secrets for accessing the data. You can always test the connection. Make sure that all services are up and running when you test the connection to them.



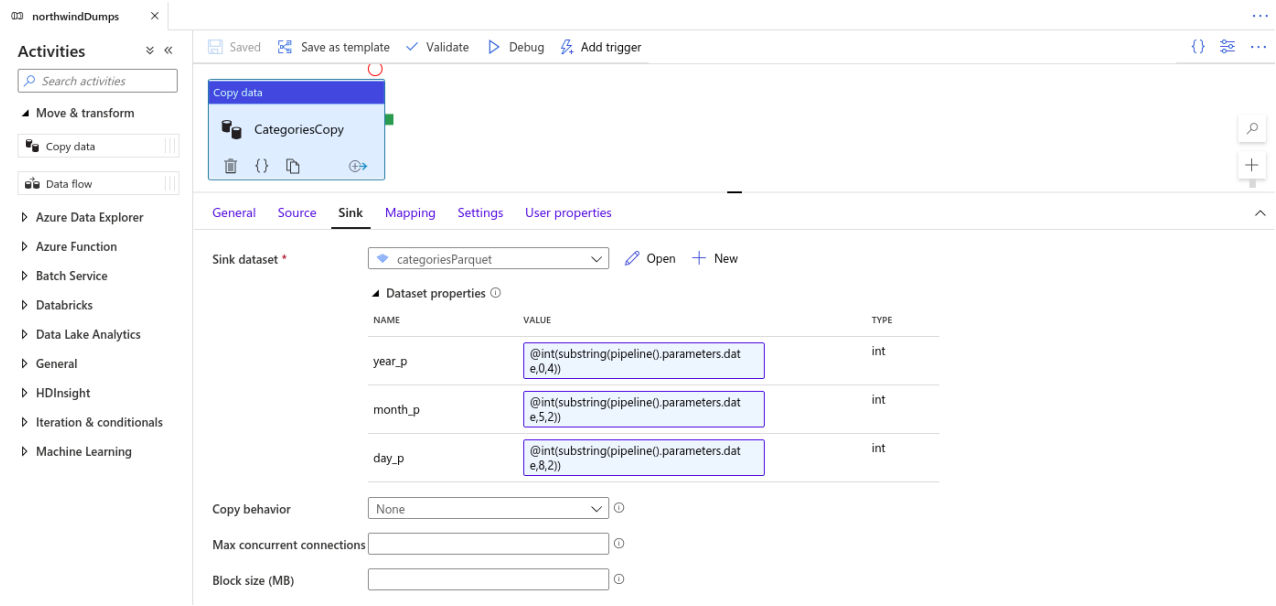
Storage Linked Services - SQL Linked Services
You probably need many Data Factoryresources, especially datasets. Therefore, make sure to put them in a nice structure. Work with folders. Give them some meaningful name. We create all datasets that represent the Data in the SQL DB tables. Furthermore, we create all datasets that represent the data in the destination (ADLS2 - parquet file). Here we define the destination Parquet dynamically. The destination folder is parametrized. This will create a kind of data partition.



Basic ETL Processing with Azure Data Factory - The Pipeline - Linked Services and Datasets

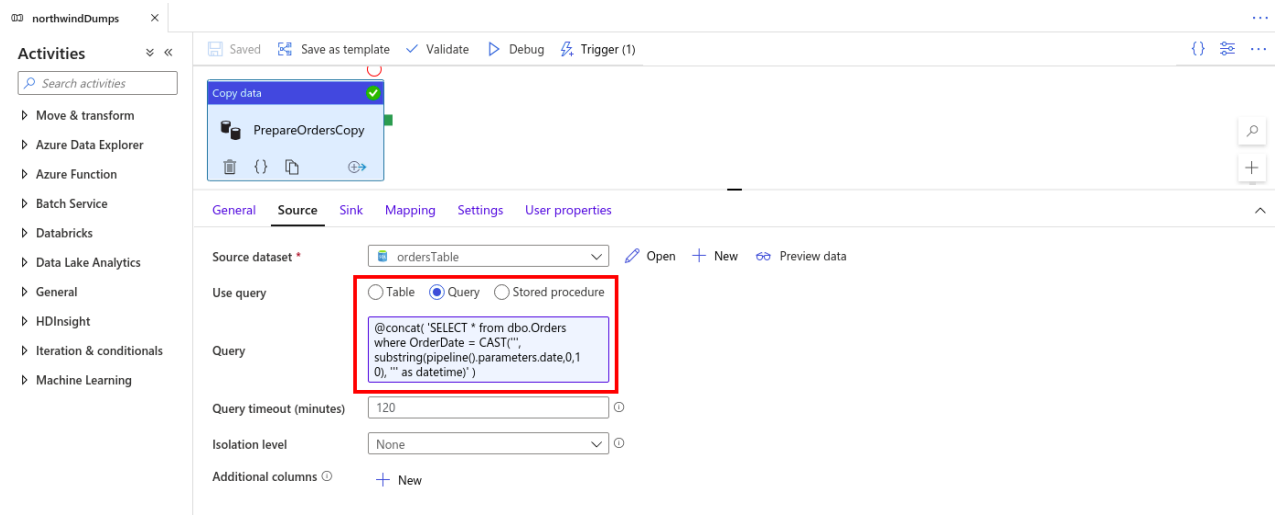
2.1.2. The Copy Activities

The copy activity copies data from a source to a sink dataset. Furthermore, it allows basic mapping (no conditional mapping) such as renaming or typecasting. There are various data sources allowed such as Azure Blob storage, Amazon Redshift, or S3, whereas sinks are more or less limited to the Azure storage systems such as Azure SQL, Azure Synapse, or Azure Blob Storage. In our scenario, we specify a copy activity for each table with the according to Source and Sink datasets. In the Sink dataset, we specify the dataset properties to include the year, month, day parameters.



The Copy Activities - I

However, when creating the Orders, we want to read only the data from specific order date. Therefore, we need to enter a query in the Source tab. As we need to parametrize the date, we need to add some dynamic content to the query. Here we need to escape a single quote. In Azures's dynamic expression builder this is done by another single quote.



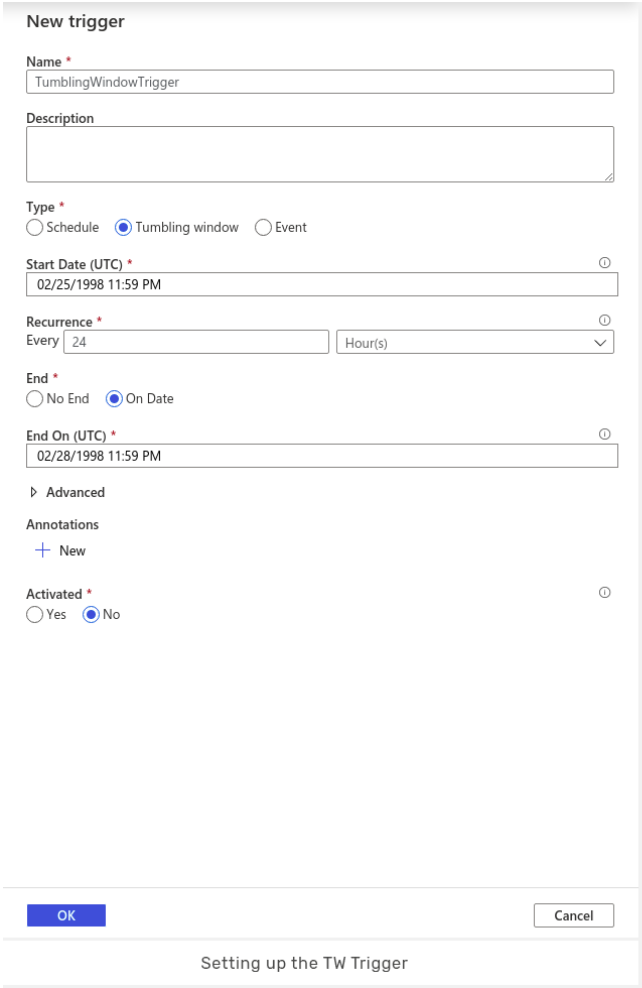
The Copy Activities II

2.1.3. The Trigger

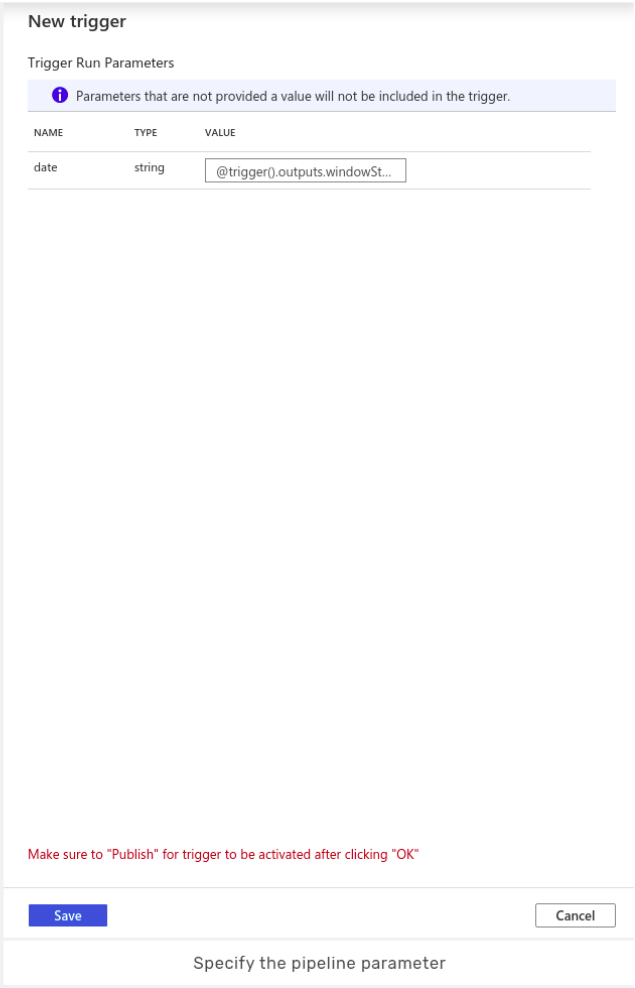
In our scenario, we want to do the dumping on a daily basis. However, as the data originates from 1996-1998, we need a backfilling mechanism. Therefore, we choose a so-called Tumbling Window (TW) Trigger.

You can find a more detailed discussion on the pros and cons of that trigger in our final article Building an Azure Data Factory Pipeline for Professional Data Warehousing. In order to create a trigger, go to your pipeline and click on the Trigger tab. Add a new Tumbling Window trigger and choose the appropriate start and end date. You can either activate it directly, which will cause that it triggers some runs directly after publishing, or

you can set it to inactive and activate it later in your "Manage" tab. If you defined the parameter "date" on the pipeline, you will be asked what value that parameter should have. Here you can enter the @trigger().outputs.windowStartTime as a dynamic content.



Setting up the TW Trigger



Specify the pipeline parameter

Basic ETL Processing with Azure Data Factory - The Trigger
If you publish these last changes you can go to the monitor tab and you will recognize that the pipeline will be triggered for the days you specified in the trigger. This completes our episode on basic processing with ADF.

Wrapping It Up

Summing it up, we have created a Data Factory together with a pipeline that contains several copy activities. We saw how to create linked services and datasets that we need for the processing. We also saw how to dynamically schedule a trigger and parametrize the pipeline in terms of which data should be read and where it is stored. In our next episode, we will learn how to set up a Synapse data warehouse with a schema that is suitable for business analytics. The last episode will present the Data Factory pipeline, which aggregates the data that is going to the data warehouse.

Sources

- Azure Docs: Data Factory overview page

- Azure Docs: Data Factory Copy Activity
- MS Azure Data Pipeline Pricing
- Azure Docs: Data Factory Pricing examples
- Azure Docs: Concepts of Linked Services
- Azure Docs: Concepts of Datasets

For the next part of the tutorial click here.