# Dynamic data driven Data Factory Pipeline

## DataSets

This pipeline has 3 Datasets.

1. Control Table. This is the table where the metadata is held.
2. Source Table. This is dynamically set.
3. Target Table. This is dynamically set.

### Control Table

#### Schema

CREATE TABLE dbo.TablesToCopy(

SchemaName varchar(128) NOT NULL,

TableName varchar(128) NOT NULL,

TableType varchar(4) NULL,

SelectProcedure varchar(128) NULL,

UpsertProcedure varchar(128) NULL,

LoadFrequency varchar(20) NULL CONSTRAINT CK\_TablesToCopy\_LoadFrequency CHECK ((LoadFrequency='Hourly' OR LoadFrequency='Daily')),

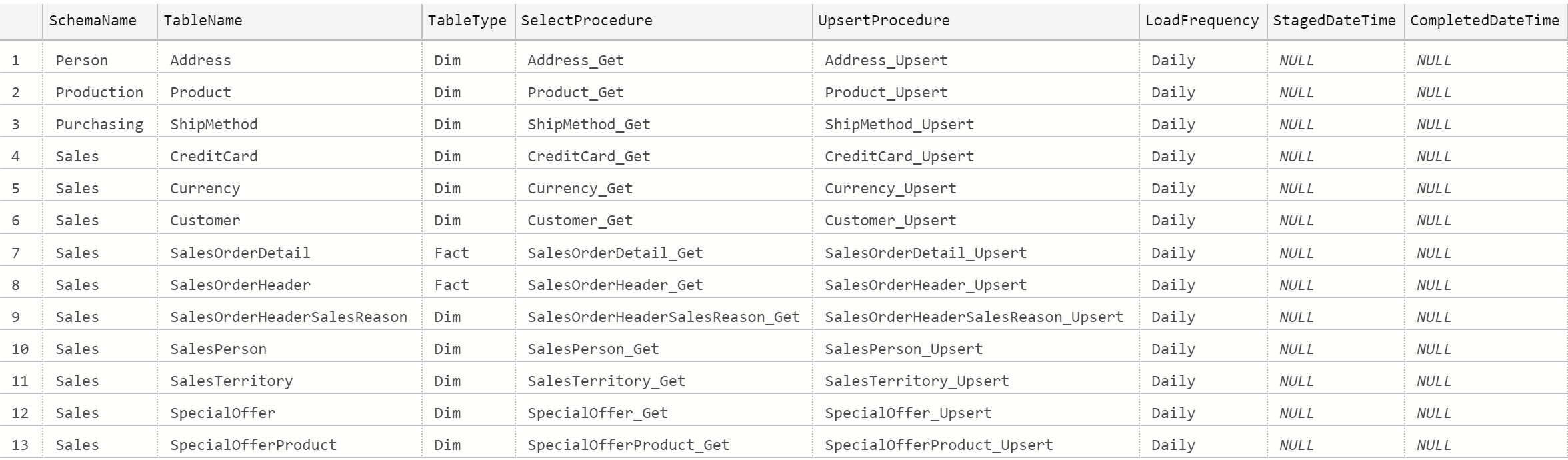
StagedDateTime datetime2(0) NULL,

CompletedDateTime datetime2(7) NULL

)

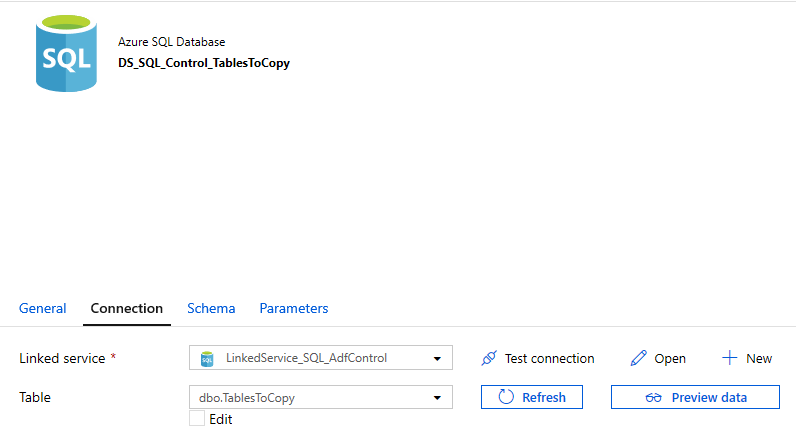
GO

#### Example Data



#### Data Connection

In this case the connection is pointed directly at the control table (dbo.TablesToCopy)



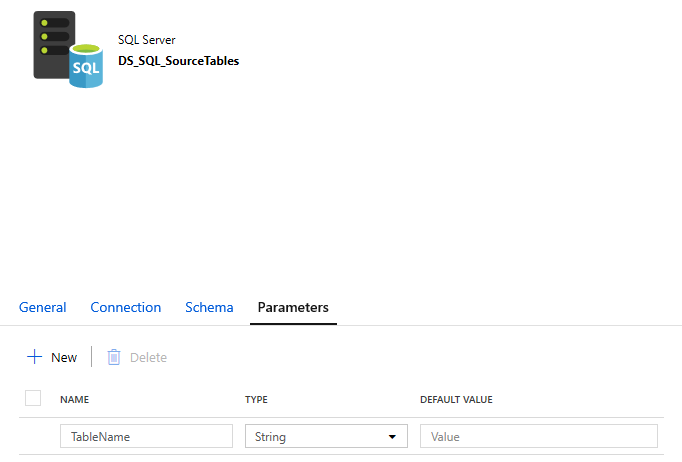
Control Table Connection

OK

### Source Tables

In this example, I’m using Adventure Works and in this case the connection string is an On Premises SQL Server. You can connect to an Azure SQL Database and the configuration will be exactly the same.

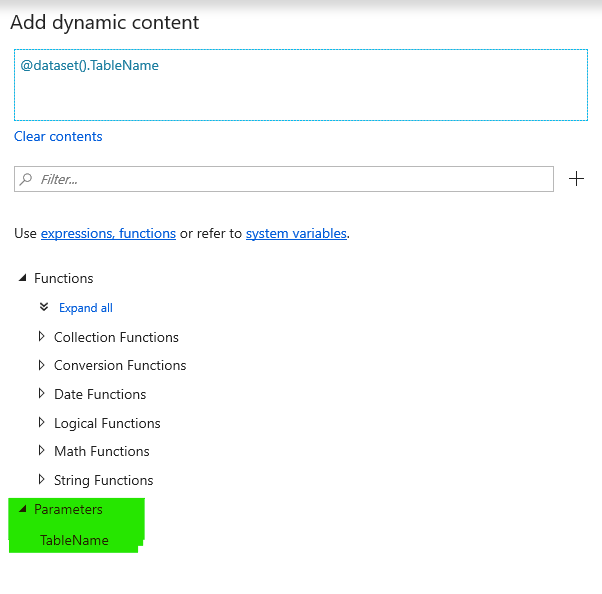
You need to add a parameter to pass the name of the table that you want to connect to. I’ve called my TableName



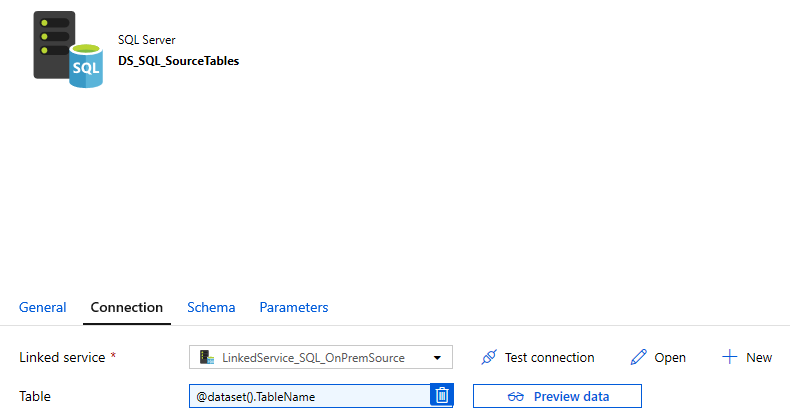
Source Table Dataset: Parameter Tab

Then in the Connection you reference the parameter using the syntax @dataset().TableName

When you click on Table field in the Connection tab, the “Add dynamic content” blade appears. In the parameter section you will see the TableName parameter you just created. Click on the parameter and it will add it for you with the right syntax.



Dynamic Content



Source Table Connection

There is no need to do anything on the Schema tab.

### Destination Table

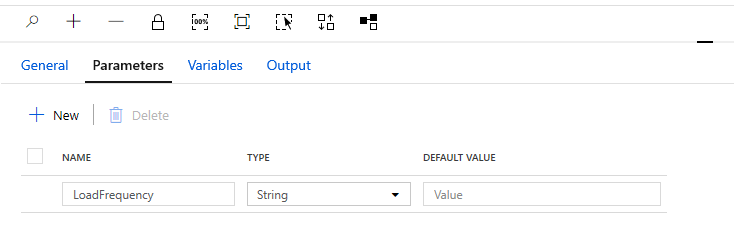
This is set up exactly the same as the source table with a Parameter configured first and the referenced in the Connection tab.

## Pipeline

A pipeline has parameters, variables and tasks.

### Parameters

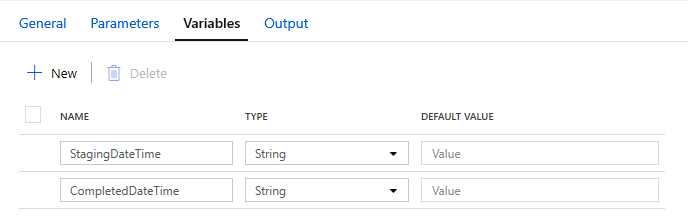
Parameters are passed to the pipeline to drive specific behaviour. In this case, the pipeline contains a parameter called LoadFrequency. This is used in the Control table query to get the list of tables. In this case there are two options, Daily and Hourly. This is completely optional and may not be required if all tables are loaded at the same frequency



Pipeline Parameters

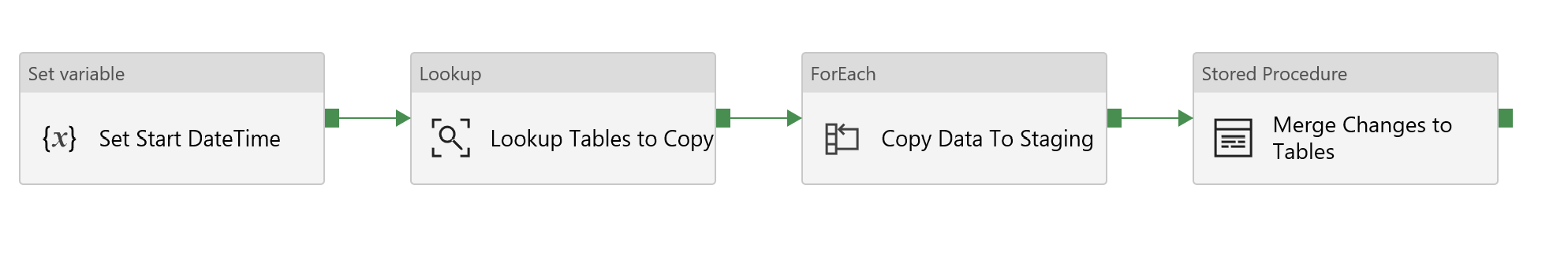
### Variables

Variables are set by ‘Set Variable’ or ‘Append Variable’ tasks within the pipeline and used in other tasks. In this case I have 2 variables, StagingDateTime and CompletedDateTime. These allow me to update the control table to say when data has reached specific target areas.



Pipeline Variables

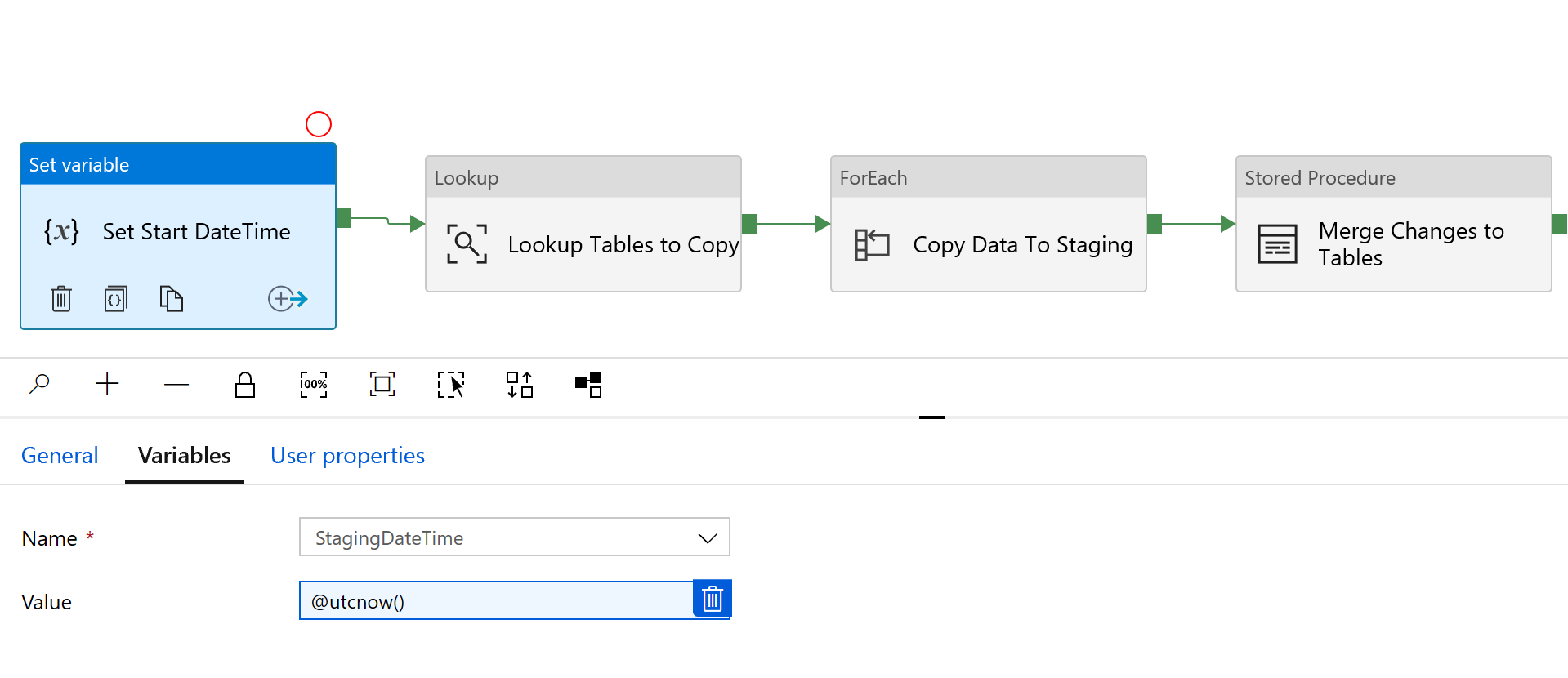
### Activities



7 Pipeline Overview

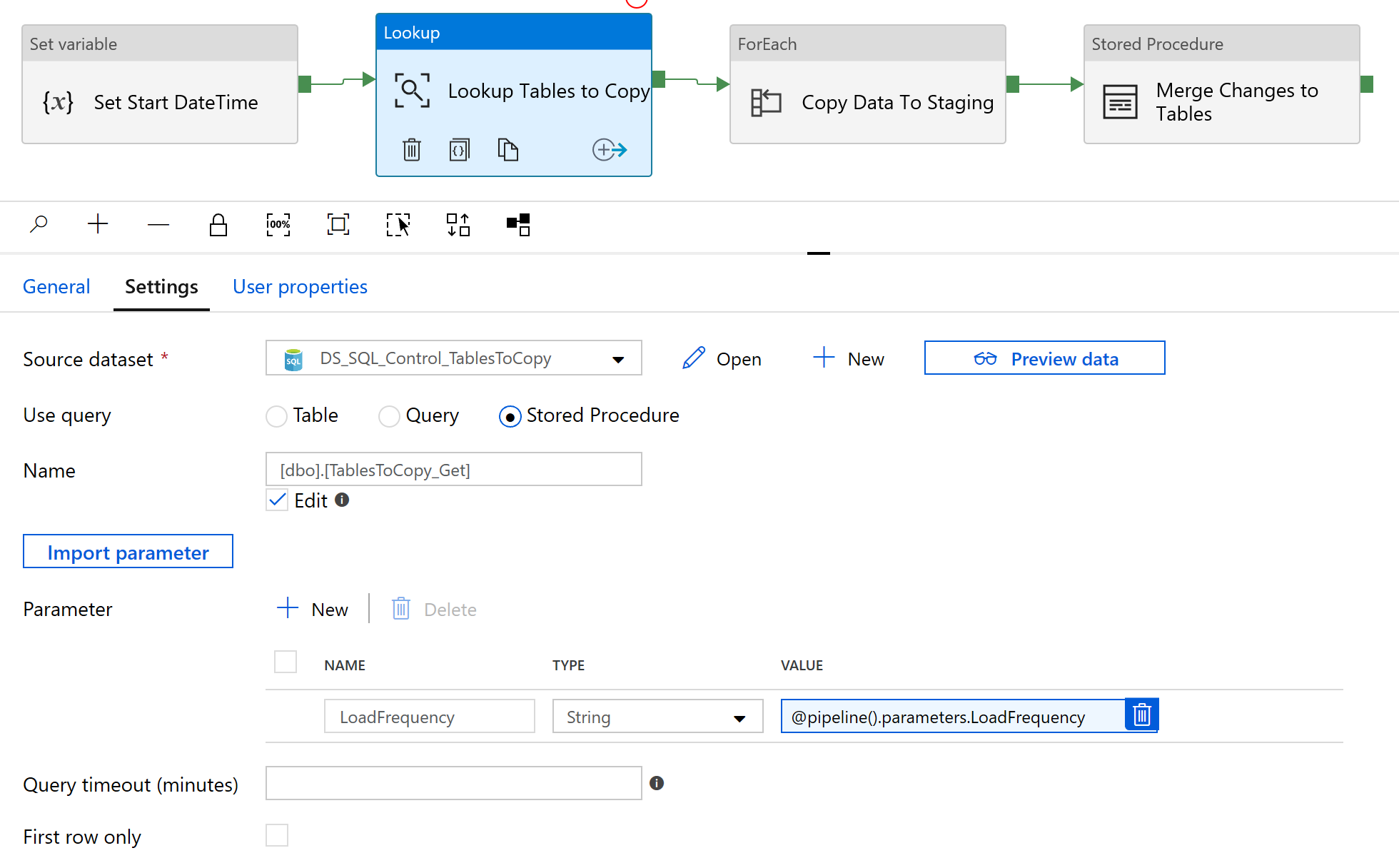
The pipeline contains the following tasks:

1. **Set variable** - sets the StagedDateTime variable with the current date and time. This uses the built in utcnow() function. If you click on the value box and open the dynamic content you will find a set of built in functions you can use. This one is in the Date set.

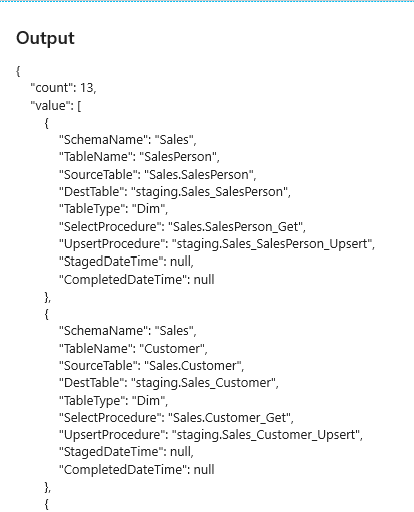


8 Set Variable to Current Date Time

1. **Lookup** – gets the list of tables in the control table that you want to copy data from. In this example I call a stored procedure, and pass in a parameter for LoadFrequency. You can also use the table directly, or a query which can also include dynamic values. This returns a json array with the data requested. Note the field names, these will be used in dynamic queries later in the pipeline.

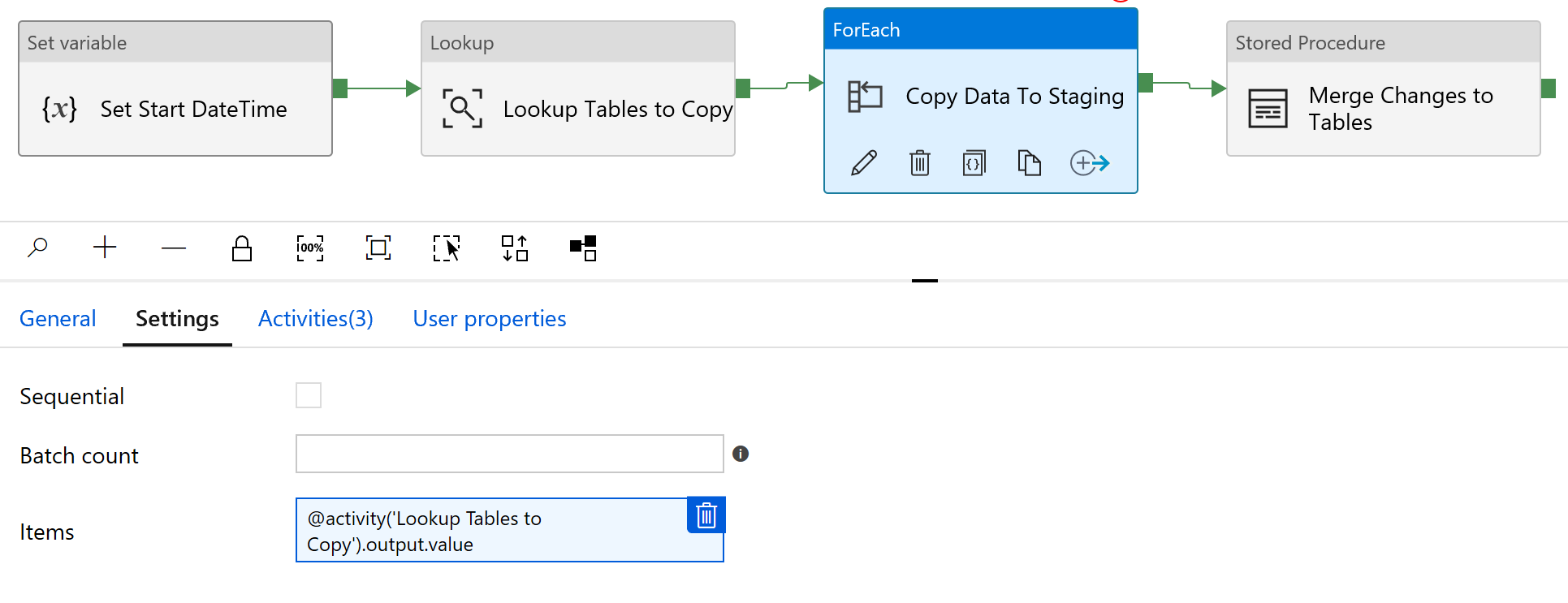


9 Get List of Tables



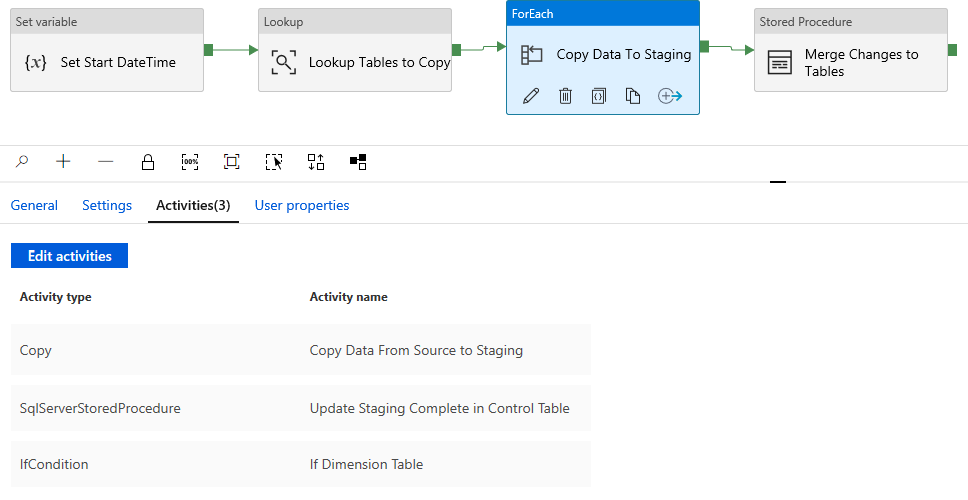
10 Json output from Lookup Activity

1. **ForEach** – iterates the list of tables and for each item. The syntax for the Items field is @activity(‘[the name of the lookup activity’).output.value.



11 For each output.value in the preceding task

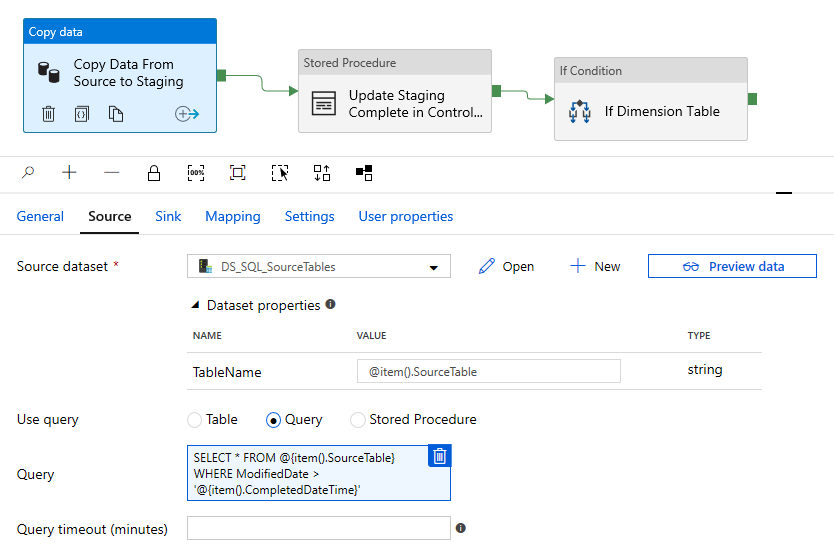
The activities tab shows the activities within the loop. Click Edit Activities to see them.



12 ForEach loop activities

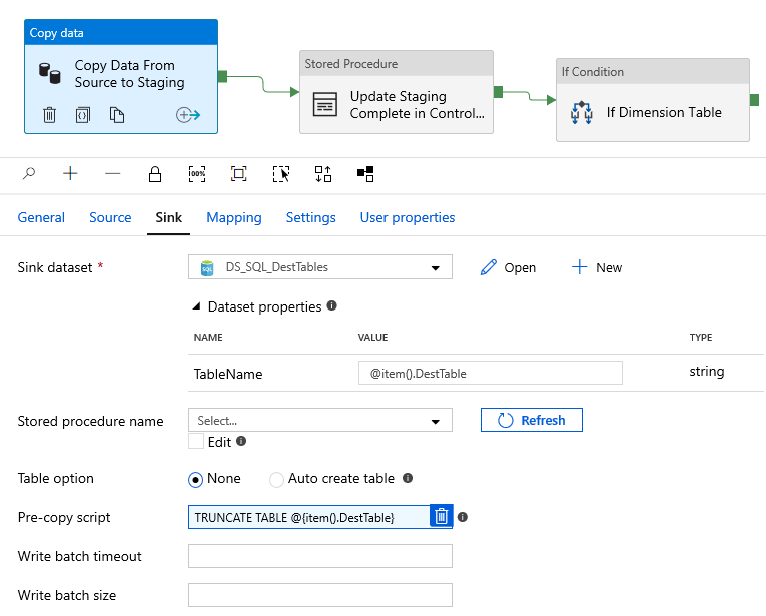
* 1. **Copy data** – copies the data from source to destination. You can choose to use a table, query, or stored procedure. In each case you can provide dynamic values. In this example I have used a query making use of the @item() properties where item is the current item within the loop.

In the Source tab we are using @{item().SourceTable} as the table to select from and @{item().CompletedDateTime} in the WHERE clause.



13 Copy Activity Source

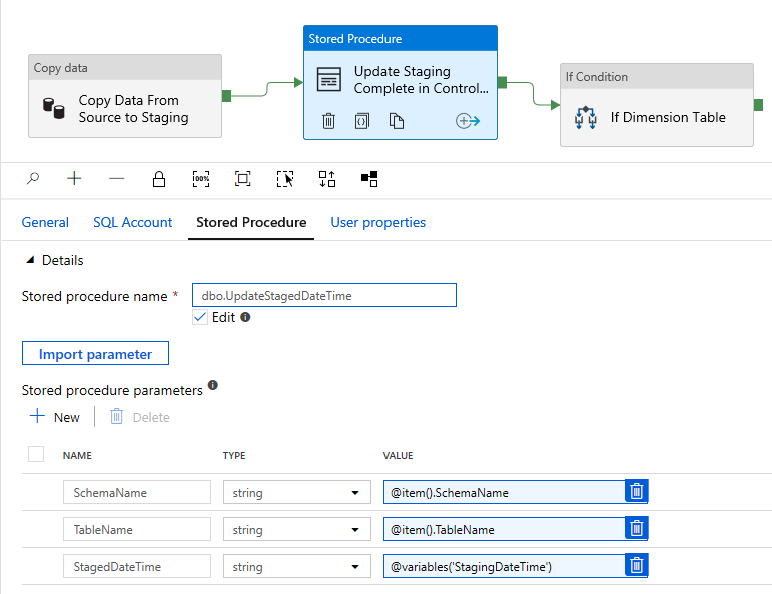
In the sink tab, we use @{item().DestTable}. Dynamic content is also supported in the pre-copy script, in this case it is used to truncate the target table since it is a transient staging table.



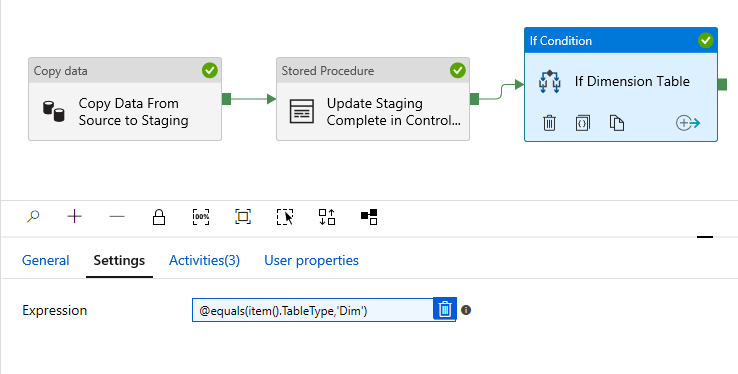
14 Copy Activity Sink

* 1. **Stored Procedure** – updates the control table to set the StagedDateTime from the value in the StagingDateTime variable. The SQL Account is the Linked Service for the control table.

The stored procedure being called has 3 parameters and these can accept dynamic values, in this case the SchemaName and TableName from the current item, and the StagedDateTime from the pipeline variable

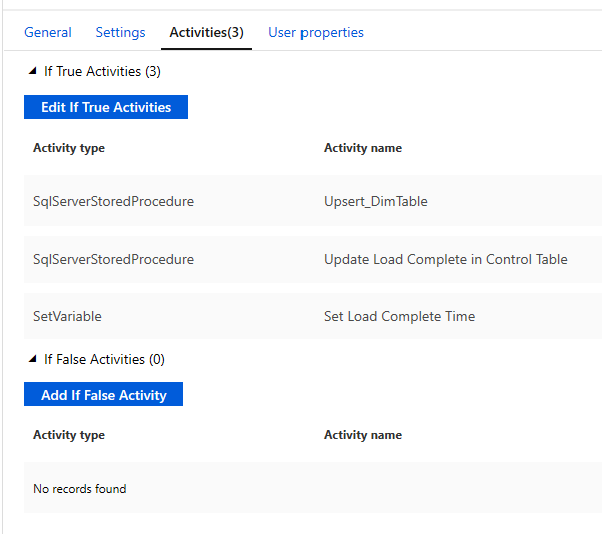


15 Stored Procedure with dynamic Parameters

* 1. **If Condition** - checks the TableType for the current item to see if this is a Dimension table. The syntax is @equals('<object1>', '<object2>') . 

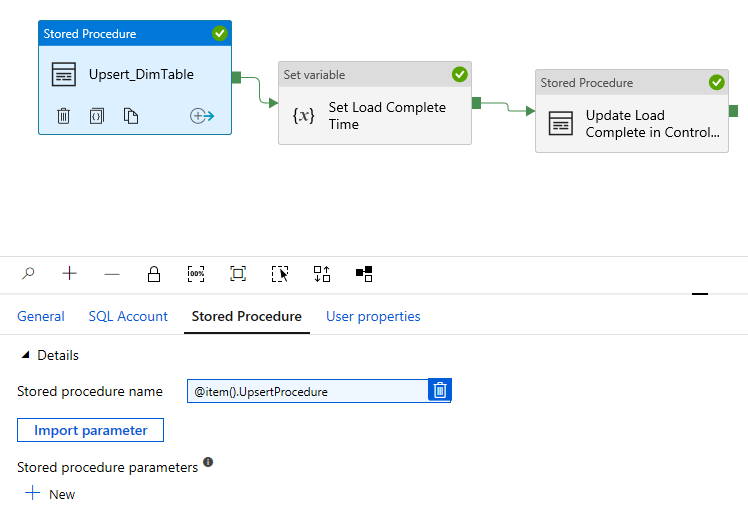
16 If Condition

The If condition allows you to specify Activities to run in True and False scenarios. In this case there are 3 activities in the True path and nothing in False



17 True and False Activities

* + 1. **Stored Procedure –** this is calling the Upsert procedure defined in the control table for the current item.



18 Dynamic Stored Procedure Name

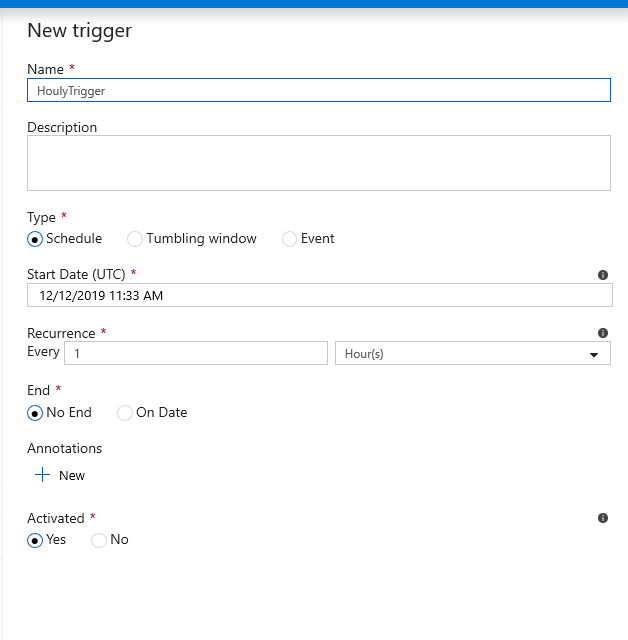
* + 1. **Set variable** – sets the CompletedDateTime with the current date and time. This is set the same way as the set variable for StagingDteTime
    2. **Stored Procedure -** updates the control table with the value from the CompletedDateTime variable. Same method as the stored procedure to update the StagedDateTime in the control table

1. **Stored Procedure** – This does any other transforms transforms. In this case I am doing the FACT table transforms since I know I have all the Dimensions already done

## Triggers

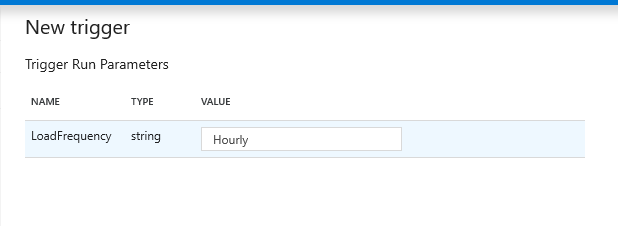
There are several ways to trigger a pipeline. For a DW load like this we use the Schedule Trigger. This allows you to define a schedule in minutes, hours, days, weeks, or months. In this example we have the option to load Hourly or Daily.

Create the first trigger for a daily load by setting the recurrence to Every 1 Hour

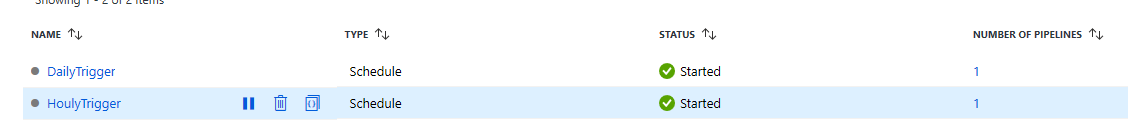


19 Create New Trigger

When you click OK, if you have parameters defined on your pipeline you will be prompted to set a value for them. In this way, you can reuse the same pipeline with different configurations. In this case we can create 2 triggers and the parameter defined in each case is used in our source query.



20 Trigger Parameter



21 Configured Triggers