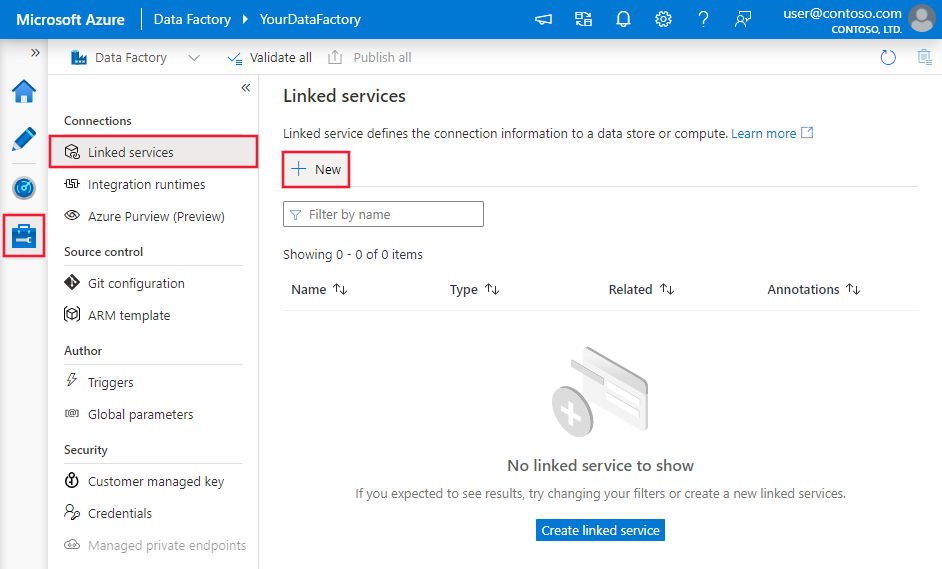
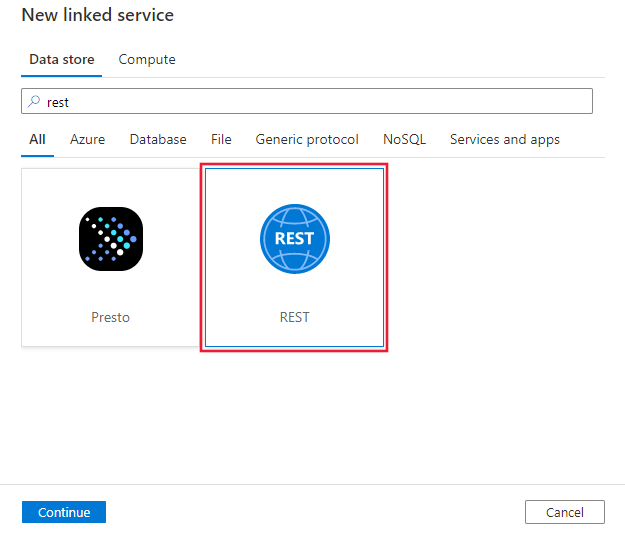
**Create a REST linked service using UI**

Use the following steps to create a REST linked service in the Azure portal UI.

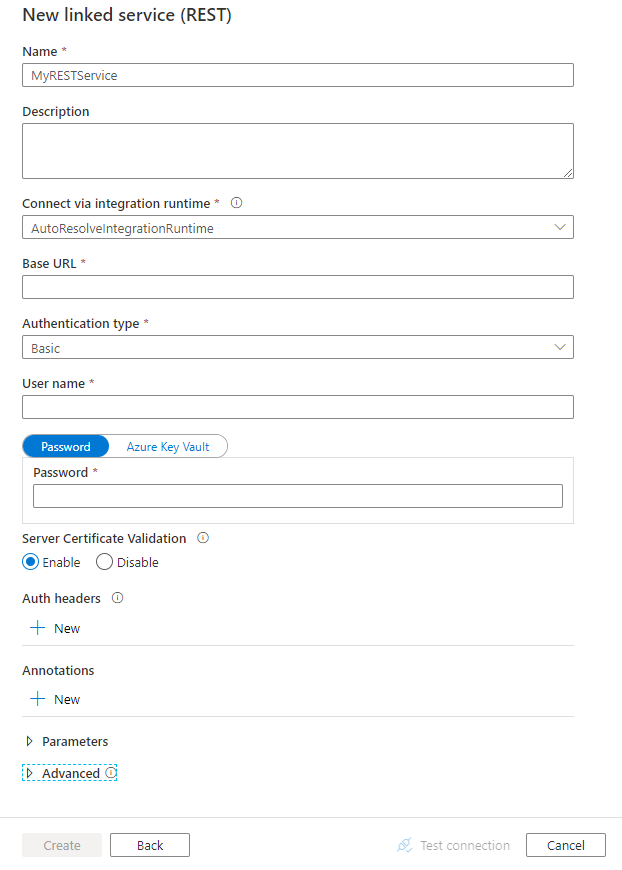
1. Browse to the Manage tab in your Azure Data Factory or Synapse workspace and select Linked Services, then click New:
   * [Azure Data Factory](https://docs.microsoft.com/en-us/azure/data-factory/connector-rest?tabs=data-factory#tabpanel_1_data-factory)
   * [Azure Synapse](https://docs.microsoft.com/en-us/azure/data-factory/connector-rest?tabs=data-factory#tabpanel_1_synapse-analytics)



1. Search for REST and select the REST connector.



1. Configure the service details, test the connection, and create the new linked service.



**Connector configuration details**

The following sections provide details about properties you can use to define Data Factory entities that are specific to the REST connector.

**Linked service properties**

**Important**

Due to Azure service security and compliance request, system-assigned managed identity authentication is no longer available in REST connector for both Copy and Mapping data flow. You are recommended to migrate existing linked services that use system-managed identity authentication to user-assigned managed identity authentication or other authentication types. Please make sure the migration to be done by **September 15, 2022**. For more detailed steps about how to create, manage user-assigned managed identities, refer to [**this**](https://docs.microsoft.com/en-us/azure/data-factory/data-factory-service-identity#user-assigned-managed-identity).

The following properties are supported for the REST linked service:

| **Property** | **Description** | **Required** |
| --- | --- | --- |
| type | The **type** property must be set to **RestService**. | Yes |
| url | The base URL of the REST service. | Yes |
| enableServerCertificateValidation | Whether to validate server-side TLS/SSL certificate when connecting to the endpoint. | No (the default is **true**) |
| authenticationType | Type of authentication used to connect to the REST service. Allowed values are **Anonymous**, **Basic**, **AadServicePrincipal**, and **ManagedServiceIdentity**. User-based OAuth isn't supported. You can additionally configure authentication headers in authHeader property. Refer to corresponding sections below on more properties and examples respectively. | Yes |
| authHeaders | Additional HTTP request headers for authentication. For example, to use API key authentication, you can select authentication type as “Anonymous” and specify API key in the header. | No |
| connectVia | The [Integration Runtime](https://docs.microsoft.com/en-us/azure/data-factory/concepts-integration-runtime) to use to connect to the data store. Learn more from [Prerequisites](https://docs.microsoft.com/en-us/azure/data-factory/connector-rest?tabs=data-factory#prerequisites) section. If not specified, this property uses the default Azure Integration Runtime. | No |

**Use basic authentication**

Set the **authenticationType** property to **Basic**. In addition to the generic properties that are described in the preceding section, specify the following properties:

| **Property** | **Description** | **Required** |
| --- | --- | --- |
| userName | The user name to use to access the REST endpoint. | Yes |
| password | The password for the user (the **userName** value). Mark this field as a **SecureString** type to store it securely in Data Factory. You can also [reference a secret stored in Azure Key Vault](https://docs.microsoft.com/en-us/azure/data-factory/store-credentials-in-key-vault). | Yes |

**Example**

JSONCopy

{

"name": "RESTLinkedService",

"properties": {

"type": "RestService",

"typeProperties": {

"authenticationType": "Basic",

"url" : "<REST endpoint>",

"userName": "<user name>",

"password": {

"type": "SecureString",

"value": "<password>"

}

},

"connectVia": {

"referenceName": "<name of Integration Runtime>",

"type": "IntegrationRuntimeReference"

}

}

}

**Use AAD service principal authentication**

Set the **authenticationType** property to **AadServicePrincipal**. In addition to the generic properties that are described in the preceding section, specify the following properties:

| **Property** | **Description** | **Required** |
| --- | --- | --- |
| servicePrincipalId | Specify the Azure Active Directory application's client ID. | Yes |
| servicePrincipalKey | Specify the Azure Active Directory application's key. Mark this field as a **SecureString** to store it securely in Data Factory, or [reference a secret stored in Azure Key Vault](https://docs.microsoft.com/en-us/azure/data-factory/store-credentials-in-key-vault). | Yes |
| tenant | Specify the tenant information (domain name or tenant ID) under which your application resides. Retrieve it by hovering the mouse in the top-right corner of the Azure portal. | Yes |
| aadResourceId | Specify the AAD resource you are requesting for authorization, for example, https://management.core.windows.net. | Yes |
| azureCloudType | For service principal authentication, specify the type of Azure cloud environment to which your AAD application is registered. Allowed values are **AzurePublic**, **AzureChina**, **AzureUsGovernment**, and **AzureGermany**. By default, the data factory's cloud environment is used. | No |

**Example**

JSONCopy

{

"name": "RESTLinkedService",

"properties": {

"type": "RestService",

"typeProperties": {

"url": "<REST endpoint e.g. https://www.example.com/>",

"authenticationType": "AadServicePrincipal",

"servicePrincipalId": "<service principal id>",

"servicePrincipalKey": {

"value": "<service principal key>",

"type": "SecureString"

},

"tenant": "<tenant info, e.g. microsoft.onmicrosoft.com>",

"aadResourceId": "<AAD resource URL e.g. https://management.core.windows.net>"

},

"connectVia": {

"referenceName": "<name of Integration Runtime>",

"type": "IntegrationRuntimeReference"

}

}

}

**Use user-assigned managed identity authentication**

Set the **authenticationType** property to **ManagedServiceIdentity**. In addition to the generic properties that are described in the preceding section, specify the following properties:

| **Property** | **Description** | **Required** |
| --- | --- | --- |
| aadResourceId | Specify the AAD resource you are requesting for authorization, for example, https://management.core.windows.net. | Yes |
| credentials | Specify the user-assigned managed identity as the credential object. | Yes |

**Example**

JSONCopy

{

"name": "RESTLinkedService",

"properties": {

"type": "RestService",

"typeProperties": {

"url": "<REST endpoint e.g. https://www.example.com/>",

"authenticationType": "ManagedServiceIdentity",

"aadResourceId": "<AAD resource URL e.g. https://management.core.windows.net>",

"credential": {

"referenceName": "credential1",

"type": "CredentialReference"

}

},

"connectVia": {

"referenceName": "<name of Integration Runtime>",

"type": "IntegrationRuntimeReference"

}

}

}

**Using authentication headers**

In addition, you can configure request headers for authentication along with the built-in authentication types.

**Example: Using API key authentication**

JSONCopy

{

"name": "RESTLinkedService",

"properties": {

"type": "RestService",

"typeProperties": {

"url": "<REST endpoint>",

"authenticationType": "Anonymous",

"authHeader": {

"x-api-key": {

"type": "SecureString",

"value": "<API key>"

}

}

},

"connectVia": {

"referenceName": "<name of Integration Runtime>",

"type": "IntegrationRuntimeReference"

}

}

}

**Dataset properties**

This section provides a list of properties that the REST dataset supports.

For a full list of sections and properties that are available for defining datasets, see [Datasets and linked services](https://docs.microsoft.com/en-us/azure/data-factory/concepts-datasets-linked-services).

To copy data from REST, the following properties are supported:

| **Property** | **Description** | **Required** |
| --- | --- | --- |
| type | The **type** property of the dataset must be set to **RestResource**. | Yes |
| relativeUrl | A relative URL to the resource that contains the data. When this property isn't specified, only the URL that's specified in the linked service definition is used. The HTTP connector copies data from the combined URL: [URL specified in linked service]/[relative URL specified in dataset]. | No |

If you were setting requestMethod, additionalHeaders, requestBody and paginationRules in dataset, it is still supported as-is, while you are suggested to use the new model in activity going forward.

**Example:**

JSONCopy

{

"name": "RESTDataset",

"properties": {

"type": "RestResource",

"typeProperties": {

"relativeUrl": "<relative url>"

},

"schema": [],

"linkedServiceName": {

"referenceName": "<REST linked service name>",

"type": "LinkedServiceReference"

}

}

}

**Copy Activity properties**

This section provides a list of properties supported by the REST source and sink.

For a full list of sections and properties that are available for defining activities, see [Pipelines](https://docs.microsoft.com/en-us/azure/data-factory/concepts-pipelines-activities).

**REST as source**

The following properties are supported in the copy activity **source** section:

| **Property** | **Description** | **Required** |
| --- | --- | --- |
| type | The **type** property of the copy activity source must be set to **RestSource**. | Yes |
| requestMethod | The HTTP method. Allowed values are **GET** (default) and **POST**. | No |
| additionalHeaders | Additional HTTP request headers. | No |
| requestBody | The body for the HTTP request. | No |
| paginationRules | The pagination rules to compose next page requests. Refer to [pagination support](https://docs.microsoft.com/en-us/azure/data-factory/connector-rest?tabs=data-factory#pagination-support) section on details. | No |
| httpRequestTimeout | The timeout (the **TimeSpan** value) for the HTTP request to get a response. This value is the timeout to get a response, not the timeout to read response data. The default value is **00:01:40**. | No |
| requestInterval | The time to wait before sending the request for next page. The default value is **00:00:01** | No |

**Note**

REST connector ignores any "Accept" header specified in additionalHeaders. As REST connector only support response in JSON, it will auto generate a header of Accept: application/json.

**Example 1: Using the Get method with pagination**

JSONCopy

"activities":[

{

"name": "CopyFromREST",

"type": "Copy",

"inputs": [

{

"referenceName": "<REST input dataset name>",

"type": "DatasetReference"

}

],

"outputs": [

{

"referenceName": "<output dataset name>",

"type": "DatasetReference"

}

],

"typeProperties": {

"source": {

"type": "RestSource",

"additionalHeaders": {

"x-user-defined": "helloworld"

},

"paginationRules": {

"AbsoluteUrl": "$.paging.next"

},

"httpRequestTimeout": "00:01:00"

},

"sink": {

"type": "<sink type>"

}

}

}

]

**Example 2: Using the Post method**

JSONCopy

"activities":[

{

"name": "CopyFromREST",

"type": "Copy",

"inputs": [

{

"referenceName": "<REST input dataset name>",

"type": "DatasetReference"

}

],

"outputs": [

{

"referenceName": "<output dataset name>",

"type": "DatasetReference"

}

],

"typeProperties": {

"source": {

"type": "RestSource",

"requestMethod": "Post",

"requestBody": "<body for POST REST request>",

"httpRequestTimeout": "00:01:00"

},

"sink": {

"type": "<sink type>"

}

}

}

]

**REST as sink**

The following properties are supported in the copy activity **sink** section:

| **Property** | **Description** | **Required** |
| --- | --- | --- |
| type | The **type** property of the copy activity sink must be set to **RestSink**. | Yes |
| requestMethod | The HTTP method. Allowed values are **POST** (default), **PUT**, and **PATCH**. | No |
| additionalHeaders | Additional HTTP request headers. | No |
| httpRequestTimeout | The timeout (the **TimeSpan** value) for the HTTP request to get a response. This value is the timeout to get a response, not the timeout to write the data. The default value is **00:01:40**. | No |
| requestInterval | The interval time between different requests in millisecond. Request interval value should be a number between [10, 60000]. | No |
| httpCompressionType | HTTP compression type to use while sending data with Optimal Compression Level. Allowed values are **none** and **gzip**. | No |
| writeBatchSize | Number of records to write to the REST sink per batch. The default value is 10000. | No |

REST connector as sink works with the REST APIs that accept JSON. The data will be sent in JSON with the following pattern. As needed, you can use the copy activity [schema mapping](https://docs.microsoft.com/en-us/azure/data-factory/copy-activity-schema-and-type-mapping#schema-mapping) to reshape the source data to conform to the expected payload by the REST API.

JSONCopy

[

{ <data object> },

{ <data object> },

...

]

**Example:**

JSONCopy

"activities":[

{

"name": "CopyToREST",

"type": "Copy",

"inputs": [

{

"referenceName": "<input dataset name>",

"type": "DatasetReference"

}

],

"outputs": [

{

"referenceName": "<REST output dataset name>",

"type": "DatasetReference"

}

],

"typeProperties": {

"source": {

"type": "<source type>"

},

"sink": {

"type": "RestSink",

"requestMethod": "POST",

"httpRequestTimeout": "00:01:40",

"requestInterval": 10,

"writeBatchSize": 10000,

"httpCompressionType": "none",

},

}

}

]

**Mapping data flow properties**

REST is supported in data flows for both integration datasets and inline datasets.

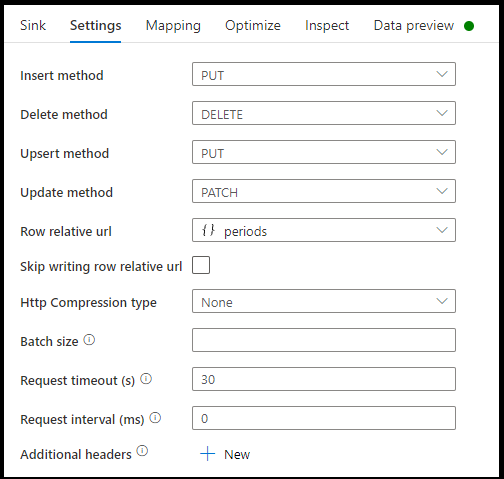
**Source transformation**

| **Property** | **Description** | **Required** |
| --- | --- | --- |
| requestMethod | The HTTP method. Allowed values are **GET** and **POST**. | Yes |
| relativeUrl | A relative URL to the resource that contains the data. When this property isn't specified, only the URL that's specified in the linked service definition is used. The HTTP connector copies data from the combined URL: [URL specified in linked service]/[relative URL specified in dataset]. | No |
| additionalHeaders | Additional HTTP request headers. | No |
| httpRequestTimeout | The timeout (the **TimeSpan** value) for the HTTP request to get a response. This value is the timeout to get a response, not the timeout to write the data. The default value is **00:01:40**. | No |
| requestInterval | The interval time between different requests in millisecond. Request interval value should be a number between [10, 60000]. | No |
| QueryParameters.*request\_query\_parameter* OR QueryParameters['request\_query\_parameter'] | "request\_query\_parameter" is user-defined, which references one query parameter name in the next HTTP request URL. | No |

**Sink transformation**

| **Property** | **Description** | **Required** |
| --- | --- | --- |
| additionalHeaders | Additional HTTP request headers. | No |
| httpRequestTimeout | The timeout (the **TimeSpan** value) for the HTTP request to get a response. This value is the timeout to get a response, not the timeout to write the data. The default value is **00:01:40**. | No |
| requestInterval | The interval time between different requests in millisecond. Request interval value should be a number between [10, 60000]. | No |
| httpCompressionType | HTTP compression type to use while sending data with Optimal Compression Level. Allowed values are **none** and **gzip**. | No |
| writeBatchSize | Number of records to write to the REST sink per batch. The default value is 10000. | No |

You can set the delete, insert, update, and upsert methods as well as the relative row data to send to the REST sink for CRUD operations.



**Sample data flow script**

Notice the use of an alter row transformation prior to the sink to instruct ADF what type of action to take with your REST sink. I.e. insert, update, upsert, delete.

Copy

AlterRow1 sink(allowSchemaDrift: true,

validateSchema: false,

deletable:true,

insertable:true,

updateable:true,

upsertable:true,

rowRelativeUrl: 'periods',

insertHttpMethod: 'PUT',

deleteHttpMethod: 'DELETE',

upsertHttpMethod: 'PUT',

updateHttpMethod: 'PATCH',

timeout: 30,

requestFormat: ['type' -> 'json'],

skipDuplicateMapInputs: true,

skipDuplicateMapOutputs: true) ~> sink1

**Pagination support**

When copying data from REST APIs, normally, the REST API limits its response payload size of a single request under a reasonable number; while to return large amount of data, it splits the result into multiple pages and requires callers to send consecutive requests to get next page of the result. Usually, the request for one page is dynamic and composed by the information returned from the response of previous page.

This generic REST connector supports the following pagination patterns:

* Next request’s absolute or relative URL = property value in current response body
* Next request’s absolute or relative URL = header value in current response headers
* Next request’s query parameter = property value in current response body
* Next request’s query parameter = header value in current response headers
* Next request’s header = property value in current response body
* Next request’s header = header value in current response headers

**Pagination rules** are defined as a dictionary in dataset, which contains one or more case-sensitive key-value pairs. The configuration will be used to generate the request starting from the second page. The connector will stop iterating when it gets HTTP status code 204 (No Content), or any of the JSONPath expressions in "paginationRules" returns null.

**Supported keys** in pagination rules:

| **Key** | **Description** |
| --- | --- |
| AbsoluteUrl | Indicates the URL to issue the next request. It can be **either absolute URL or relative URL**. |
| QueryParameters.*request\_query\_parameter* OR QueryParameters['request\_query\_parameter'] | "request\_query\_parameter" is user-defined, which references one query parameter name in the next HTTP request URL. |
| Headers.*request\_header* OR Headers['request\_header'] | "request\_header" is user-defined, which references one header name in the next HTTP request. |
| EndCondition:*end\_condition* | "end\_condition" is user-defined, which indicates the condition that will end the pagination loop in the next HTTP request. |
| MaxRequestNumber | Indicates the maximum pagination request number. Leave it as empty means no limit. |
| SupportRFC5988 | By default, this is set to true if no pagination rule is defined. You can disable this rule by setting supportRFC5988 to false or remove this property from script. |

**Supported values** in pagination rules:

| **Value** | **Description** |
| --- | --- |
| Headers.*response\_header* OR Headers['response\_header'] | "response\_header" is user-defined, which references one header name in the current HTTP response, the value of which will be used to issue next request. |
| A JSONPath expression starting with "$" (representing the root of the response body) | The response body should contain only one JSON object. The JSONPath expression should return a single primitive value, which will be used to issue next request. |

**Note**

The pagination rules in mapping data flows is different from it in copy activity in the following aspects:

1. Range is not supported in mapping data flows.
2. [''] is not supported in mapping data flows. Instead, use {} to escape special character. For example, body.{@odata.nextLink}, whose JSON node @odata.nextLink contains special character . .
3. The end condition is supported in mapping data flows, but the condition syntax is different from it in copy activity. body is used to indicate the response body instead of $. header is used to indicate the response header instead of headers. Here are two examples showing this difference:
   * Example 1:  
     Copy activity: **"EndCondition:$.data": "Empty"**  
     Mapping data flows: **"EndCondition:body.data": "Empty"**
   * Example 2:  
     Copy activity: **"EndCondition:headers.complete": "Exist"**  
     Mapping data flows: **"EndCondition:header.complete": "Exist"**

**Pagination rules examples**

This section provides a list of examples for pagination rules settings.

**Example 1: Variables in QueryParameters**

This example provides the configuration steps to send multiple requests whose variables are in QueryParameters.

**Multiple requests:**

Copy

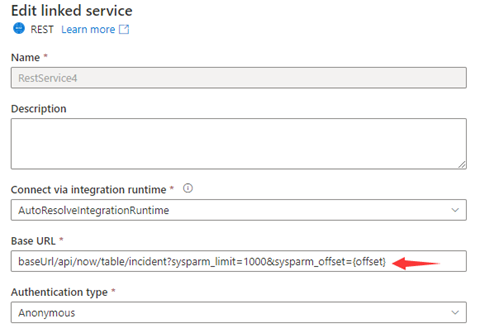
baseUrl/api/now/table/incident?sysparm\_limit=1000&sysparm\_offset=0,

baseUrl/api/now/table/incident?sysparm\_limit=1000&sysparm\_offset=1000,

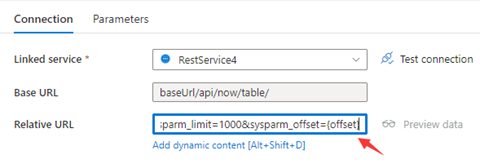
......

baseUrl/api/now/table/incident?sysparm\_limit=1000&sysparm\_offset=10000

*Step 1*: Input sysparm\_offset={offset} either in **Base URL** or **Relative URL** as shown in the following screenshots:



or



*Step 2*: Set **Pagination rules** as either option 1 or option 2：

* Option1: **"QueryParameters.{offset}" : "RANGE:0:10000:1000"**
* Option2: **"AbsoluteUrl.{offset}" : "RANGE:0:10000:1000"**

**Example 2：Variables in AbsoluteUrl**

This example provides the configuration steps to send multiple requests whose variables are in AbsoluteUrl.

**Multiple requests:**

Copy

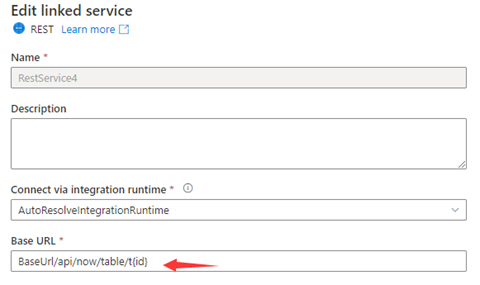
BaseUrl/api/now/table/t1

BaseUrl/api/now/table/t2

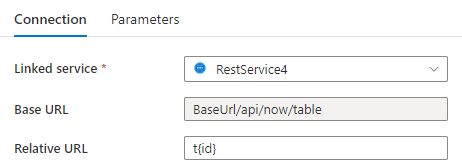
......

BaseUrl/api/now/table/t100

*Step 1*: Input {id} either in **Base URL** in the linked service configuration page or **Relative URL** in the dataset connection pane.



or



*Step 2*: Set **Pagination rules** as **"AbsoluteUrl.{id}" :"RANGE:1:100:1"**.

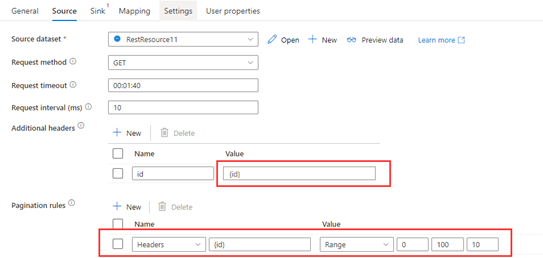
**Example 3：Variables in Headers**

This example provides the configuration steps to send multiple requests whose variables are in Headers.

**Multiple requests:**  
RequestUrl: *https://example/table*  
Request 1: Header(id->0)  
Request 2: Header(id->10)  
......  
Request 100: Header(id->100)

*Step 1*: Input {id} in **Additional headers**.

*Step 2*: Set **Pagination rules** as **"Headers.{id}" : "RARNGE:0:100:10"**.



**Example 4：Variables are in AbsoluteUrl/QueryParameters/Headers, the end variable is not pre-defined and the end condition is based on the response**

This example provides configuration steps to send multiple requests whose variables are in AbsoluteUrl/QueryParameters/Headers but the end variable is not defined. For different responses, different end condition rule settings are shown in Example 4.1-4.6.

**Multiple requests:**

Copy

Request 1: baseUrl/api/now/table/incident?sysparm\_limit=1000&sysparm\_offset=0,

Request 2: baseUrl/api/now/table/incident?sysparm\_limit=1000&sysparm\_offset=1000,

Request 3: baseUrl/api/now/table/incident?sysparm\_limit=1000&sysparm\_offset=2000,

......

Two responses encountered in this example:

Response 1：

JSONCopy

{

Data: [

{key1: val1, key2: val2

},

{key1: val3, key2: val4

}

]

}

Response 2：

JSONCopy

{

Data: [

{key1: val5, key2: val6

},

{key1: val7, key2: val8

}

]

}

*Step 1*: Set the range of **Pagination rules** as [Example 1](https://docs.microsoft.com/en-us/azure/data-factory/connector-rest?tabs=data-factory#example-1-variables-in-queryparameters) and leave the end of range empty as **"AbsoluteUrl.{offset}": "RANGE:0::1000"**.

*Step 2*: Set different end condition rules according to different last responses. See below examples:

* **Example 4.1: The pagination ends when the value of the specific node in response is empty**

The REST API returns the last response in the following structure:

JSONCopy

{

Data: []

}

Set the end condition rule as **"EndCondition:$.data": "Empty"** to end the pagination when the value of the specific node in response is empty.

Screenshot showing the End Condition setting for Example 4.1.

* **Example 4.2: The pagination ends when the value of the specific node in response dose not exist**

The REST API returns the last response in the following structure:

JSONCopy

{}

Set the end condition rule as **"EndCondition:$.data": "NonExist"** to end the pagination when the value of the specific node in response dose not exist.

Screenshot showing the End Condition setting for Example 4.2.

* **Example 4.3: The pagination ends when the value of the specific node in response exists**

The REST API returns the last response in the following structure:

JSONCopy

{

Data: [

{key1: val991, key2: val992

},

{key1: val993, key2: val994

}

],

Complete: true

}

Set the end condition rule as **"EndCondition:$.Complete": "Exist"** to end the pagination when the value of the specific node in response exists.

Screenshot showing the End Condition setting for Example 4.3.

* **Example 4.4: The pagination ends when the value of the specific node in response is a user-defined const value**

The REST API returns the response in the following structure:

JSONCopy

{

Data: [

{key1: val1, key2: val2

},

{key1: val3, key2: val4

}

],

Complete: false

}

......

And the last response is in the following structure:

JSONCopy

{

Data: [

{key1: val991, key2: val992

},

{key1: val993, key2: val994

}

],

Complete: true

}

Set the end condition rule as **"EndCondition:$.Complete": "Const:true"** to end the pagination when the value of the specific node in response is a user-defined const value.

Screenshot showing the End Condition setting for Example 4.4.

* **Example 4.5: The pagination ends when the value of the header key in response equals to user-defined const value**

The header keys in REST API responses are shown in the structure below:

Response header 1: header(Complete->0)  
......  
Last Response header: header(Complete->1)

Set the end condition rule as **"EndCondition:headers.Complete": "Const:1"** to end the pagination when the value of the header key in response is equal to user-defined const value.

Screenshot showing the End Condition setting for Example 4.5.

* **Example 4.6: The pagination ends when the key exists in the response header**

The header keys in REST API responses are shown in the structure below:

Response header 1: header()  
......  
Last Response header: header(CompleteTime->20220920)

Set the end condition rule as **"EndCondition:headers.CompleteTime": "Exist"** to end the pagination when the key exists in the response header.

Screenshot showing the End Condition setting for Example 4.6.

**Example 5：Set end condition to avoid endless requests when range rule is not defined**

This example provides the configuration steps to send multiple requests when the range rule is not used. The end condition can be set refer to Example 4.1-4.6 to avoid endless requests. The REST API returns response in the following structure, in which case next page's URL is represented in ***paging.next***.

JSONCopy

{

"data": [

{

"created\_time": "2017-12-12T14:12:20+0000",

"name": "album1",

"id": "1809938745705498\_1809939942372045"

},

{

"created\_time": "2017-12-12T14:14:03+0000",

"name": "album2",

"id": "1809938745705498\_1809941802371859"

},

{

"created\_time": "2017-12-12T14:14:11+0000",

"name": "album3",

"id": "1809938745705498\_1809941879038518"

}

],

"paging": {

"cursors": {

"after": "MTAxNTExOTQ1MjAwNzI5NDE=",

"before": "NDMyNzQyODI3OTQw"

},

"previous": "https://graph.facebook.com/me/albums?limit=25&before=NDMyNzQyODI3OTQw",

"next": "https://graph.facebook.com/me/albums?limit=25&after=MTAxNTExOTQ1MjAwNzI5NDE="

}

}

...

The last response is:

JSONCopy

{

"data": [],

"paging": {

"cursors": {

"after": "MTAxNTExOTQ1MjAwNzI5NDE=",

"before": "NDMyNzQyODI3OTQw"

},

"previous": "https://graph.facebook.com/me/albums?limit=25&before=NDMyNzQyODI3OTQw",

"next": "Same with Last Request URL"

}

}

*Step 1*: Set **Pagination rules** as **"AbsoluteUrl": "$.paging.next"**.

*Step 2*: If next in the last response is always same with the last request URL and not empty, endless requests will be sent. The end condition can be used to avoid endless requests. Therefore, set the end condition rule refer to Example 4.1-4.6.

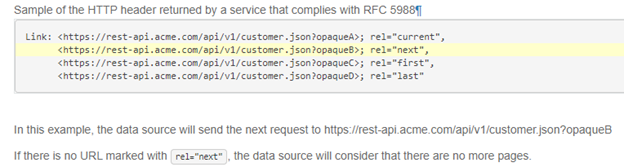
**Example 6：Set the max request number to avoid endless request**

Set **MaxRequestNumber** to avoid endless request as shown in the following screenshot:

Screenshot showing the Max Request Number setting for Example 6.

**Example 7：The RFC 5988 pagination rule is supported by default**

The backend will automatically get the next URL based on the RFC 5988 style links in the header.



**Tip**

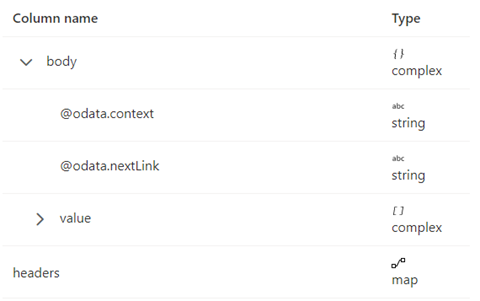
If you don't want to enable this default pagination rule, you can set supportRFC5988 to false or just delete it in the script.

Screenshot showing how to disable R F C 5988 setting for Example 7.

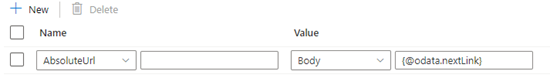
**Example 8: The next request URL is from the response body when use pagination in mapping data flows**

This example states how to set the pagination rule and the end condition rule in mapping data flows when the next request URL is from the response body.

The response schema is shown below:



The pagination rules should be set as the following screenshot:



By default, the pagination will stop when body **.{@odata.nextLink}** is null or empty.

But if the value of **@odata.nextLink** in the last response body is equal to the last request URL, then it will lead to the endless loop. To avoid this condition, define end condition rules.

* If **Value** in the last response is **Empty**, then the end condition rule can be set as below:



* If the value of the complete key in the response header equals to true indicates the end of pagination, then the end condition rule can be set as below:

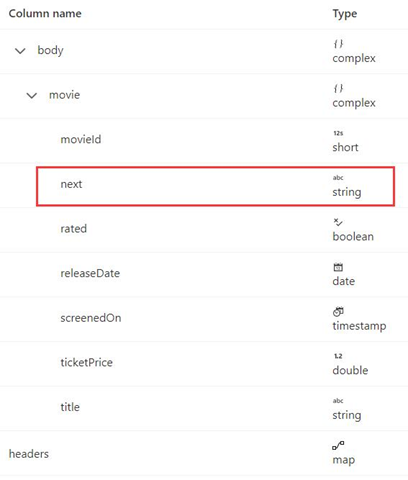


**Example 9: The response format is XML and the next request URL is from the response body when use pagination in mapping data flows**

This example states how to set the pagination rule in mapping data flows when the response format is XML and the next request URL is from the response body. As shown in the following screenshot, the first URL is *https://<user>.dfs.core.windows.net/bugfix/test/movie\_1.xml*



The response schema is shown below:



The pagination rule syntax is the same as in Example 8 and should be set as below in this example:

Screenshot showing setting the pagination rule for Example 9.

**Use OAuth**

This section describes how to use a solution template to copy data from REST connector into Azure Data Lake Storage in JSON format using OAuth.

**About the solution template**

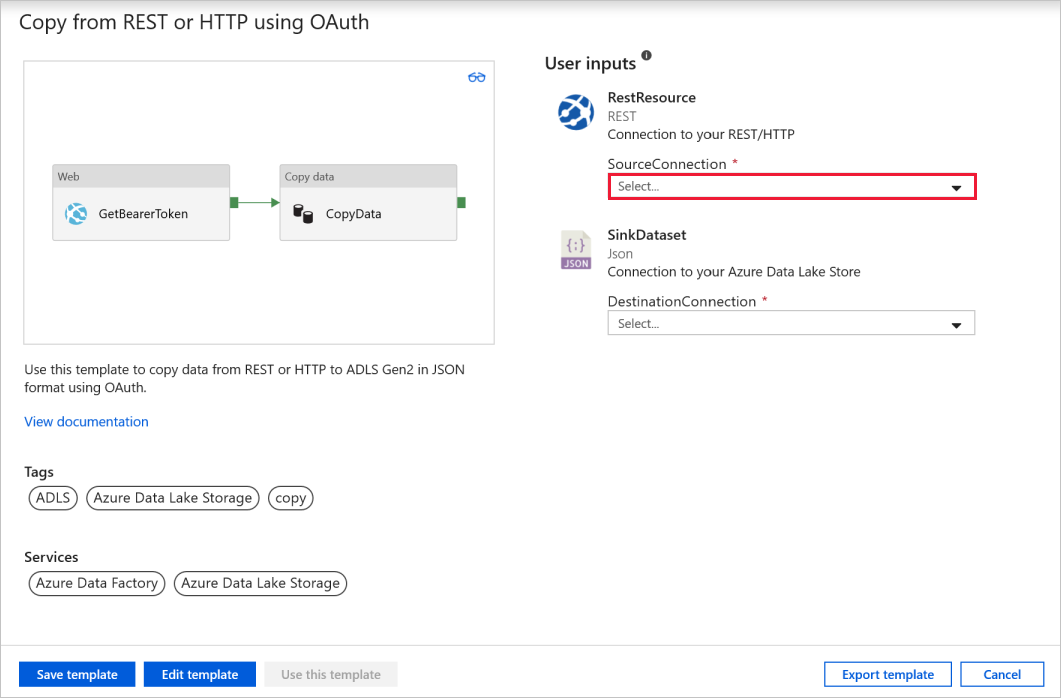
The template contains two activities:

* **Web** activity retrieves the bearer token and then pass it to subsequent Copy activity as authorization.
* **Copy** activity copies data from REST to Azure Data Lake Storage.

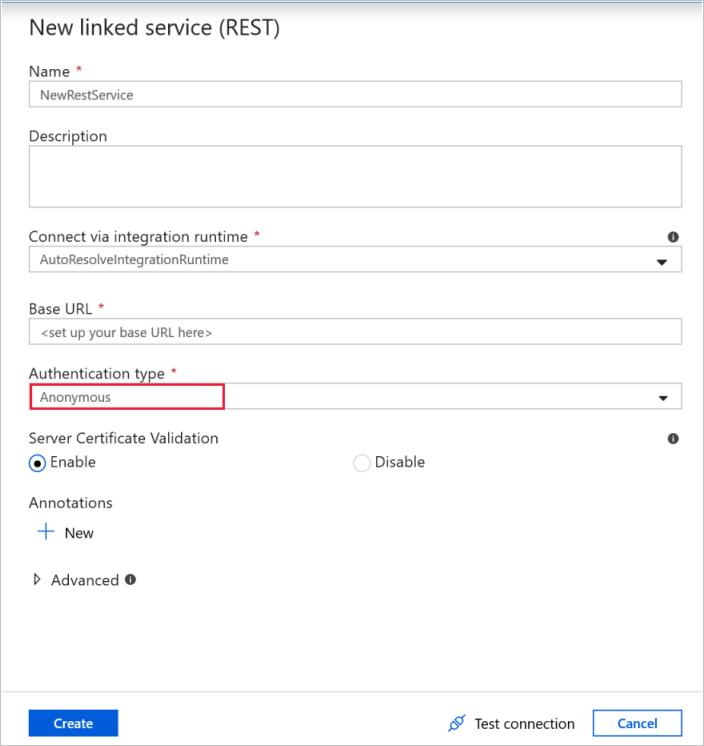
The template defines two parameters:

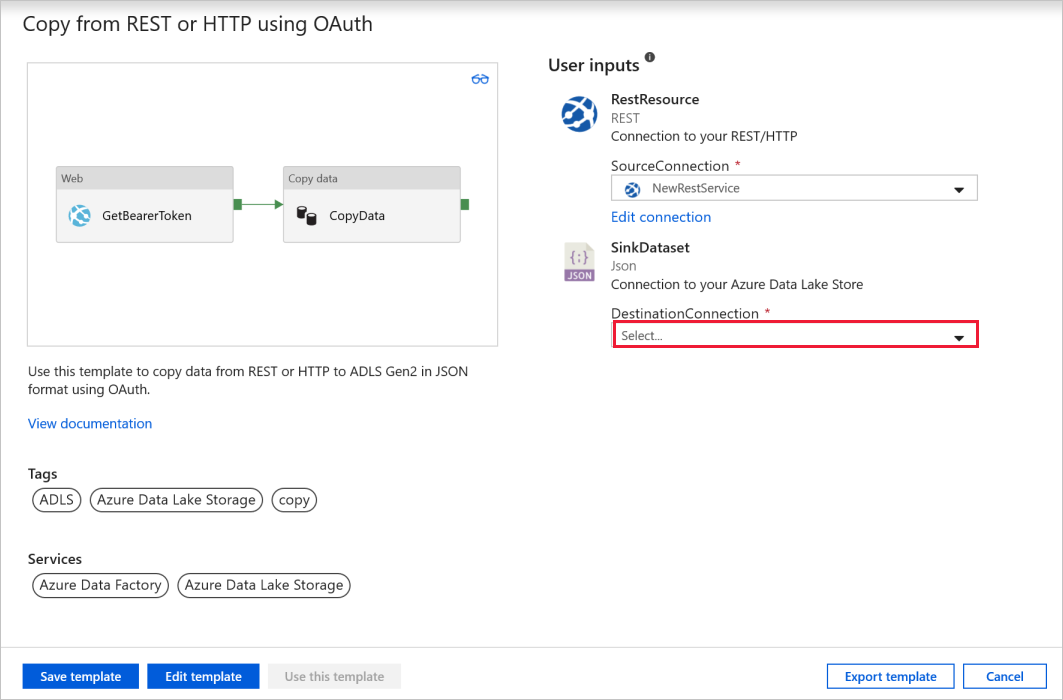
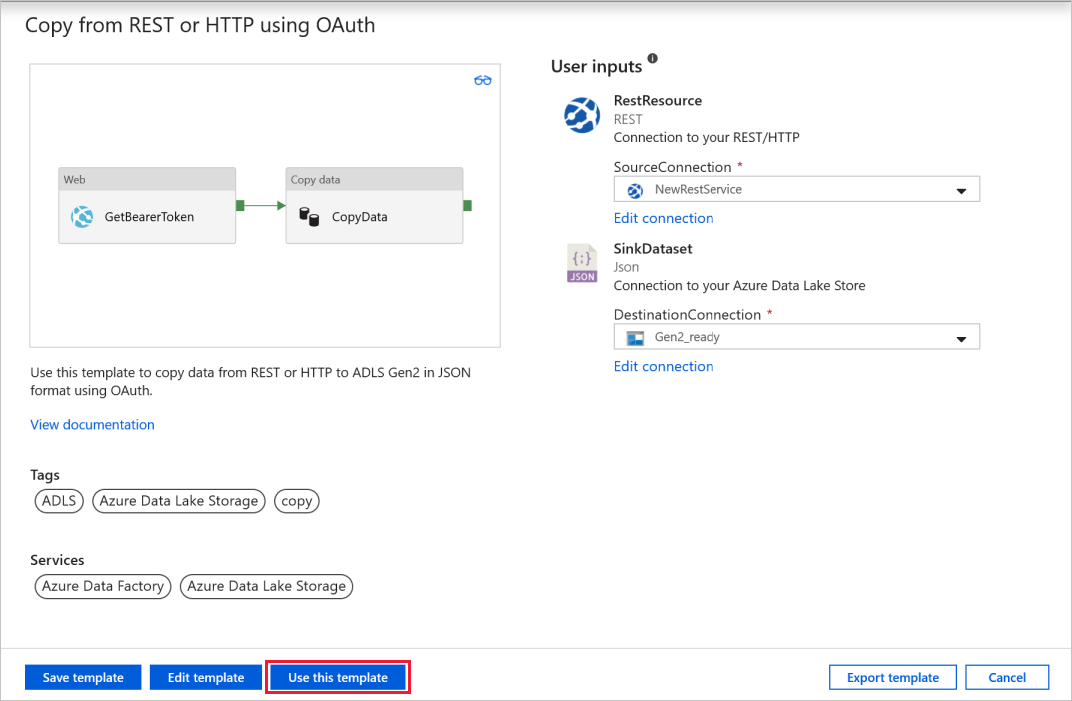
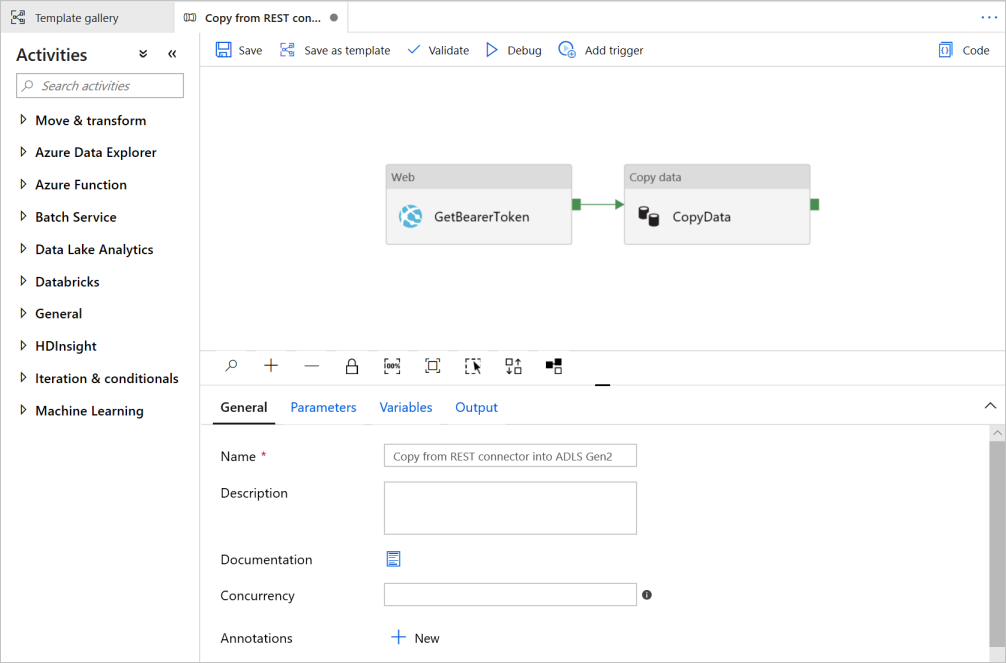
* **SinkContainer** is the root folder path where the data is copied to in your Azure Data Lake Storage.
* **SinkDirectory** is the directory path under the root where the data is copied to in your Azure Data Lake Storage.

**How to use this solution template**

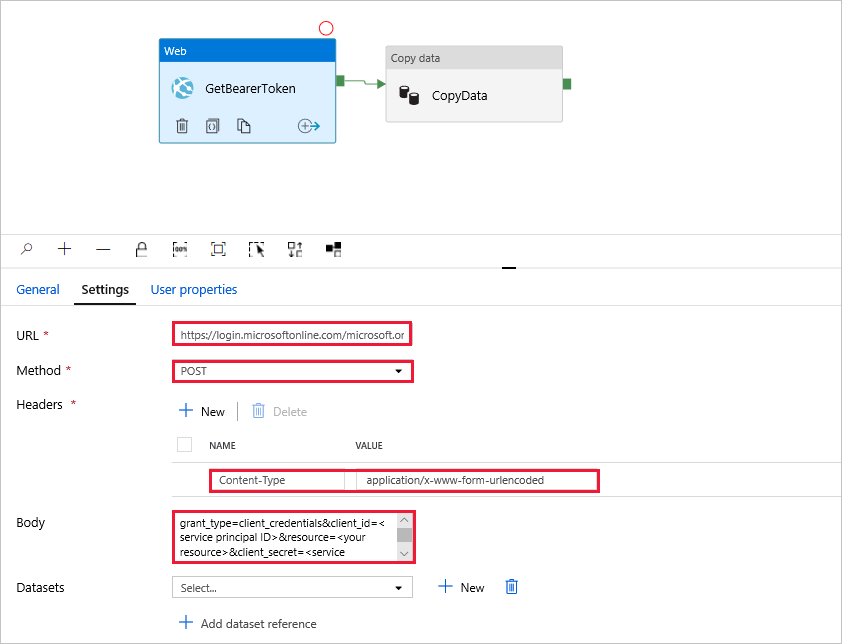
1. Go to the **Copy from REST or HTTP using OAuth** template. Create a new connection for Source Connection. 

Below are key steps for new linked service (REST) settings:

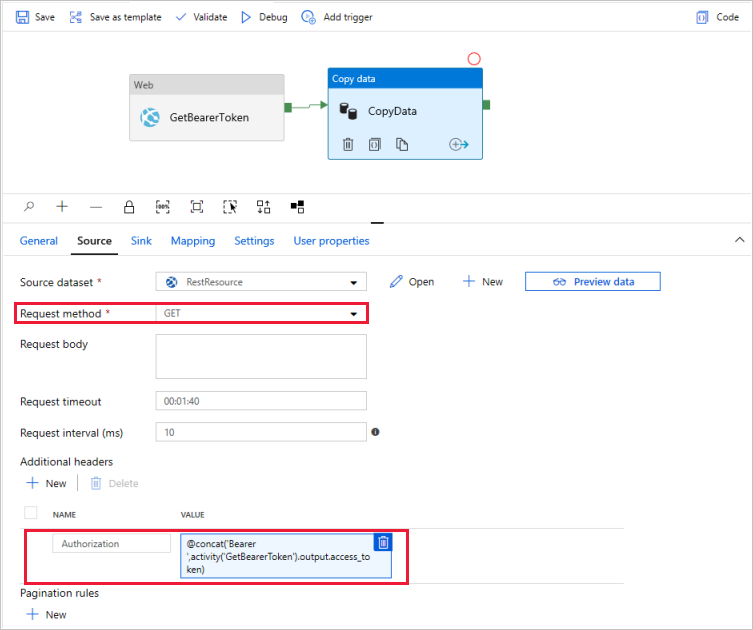
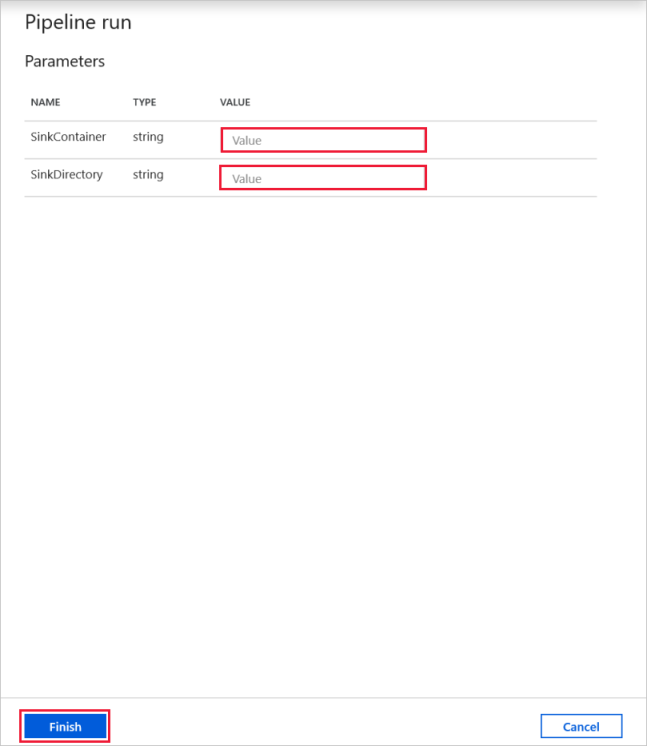
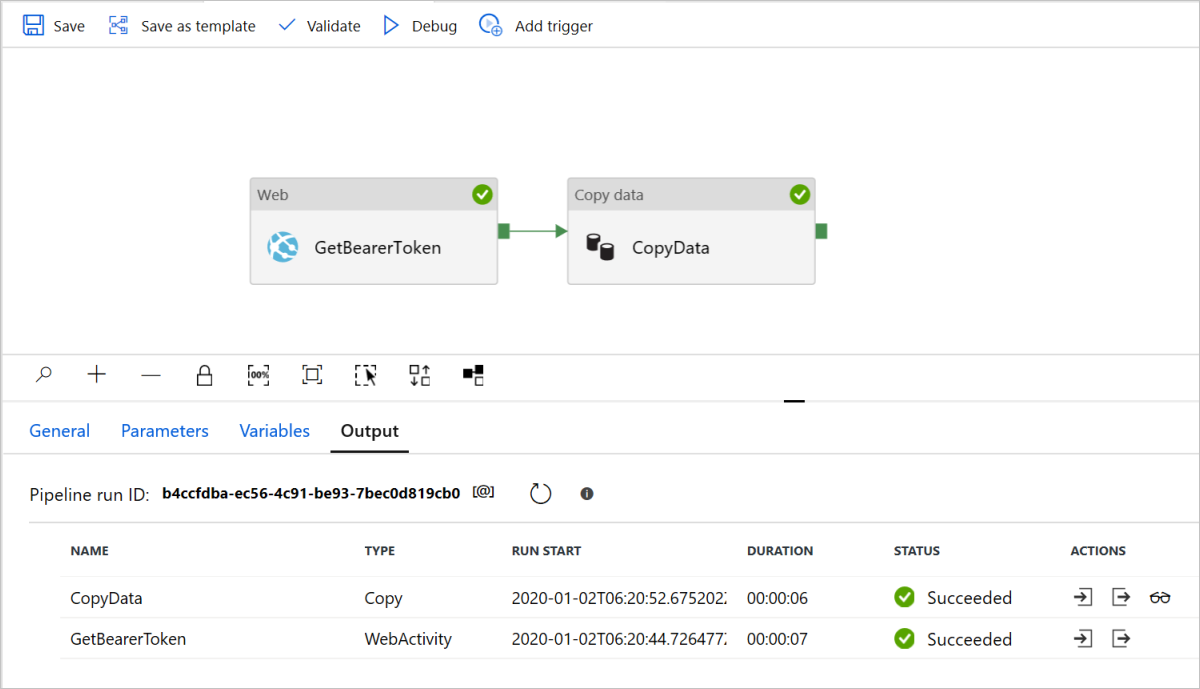
* 1. Under **Base URL**, specify the url parameter for your own source REST service.
  2. For **Authentication type**, choose *Anonymous*. 

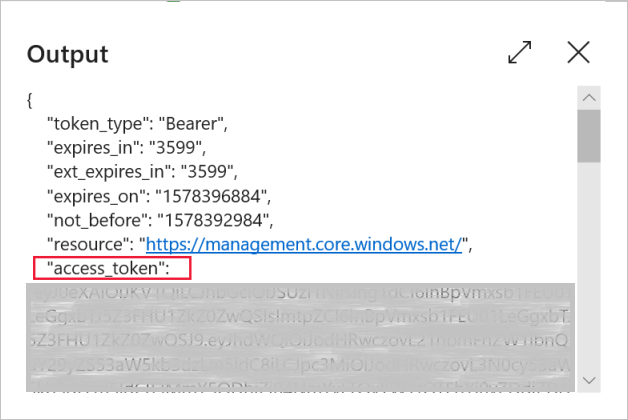
1. Create a new connection for Destination Connection.  
   
2. Select **Use this template**. 
3. You would see the pipeline created as shown in the following example: 
4. Select **Web** activity. In **Settings**, specify the corresponding **URL**, **Method**, **Headers**, and **Body** to retrieve OAuth bearer token from the login API of the service that you want to copy data from. The placeholder in the template showcases a sample of Azure Active Directory (AAD) OAuth. Note AAD authentication is natively supported by REST connector, here is just an example for OAuth flow.

| **Property** | **Description** |
| --- | --- |
| URL | Specify the url to retrieve OAuth bearer token from. for example, in the sample here it's <https://login.microsoftonline.com/microsoft.onmicrosoft.com/oauth2/token> |
| Method | The HTTP method. Allowed values are **Post** and **Get**. |
| Headers | Header is user-defined, which references one header name in the HTTP request. |
| Body | The body for the HTTP request. |

1. 
2. In **Copy data** activity, select *Source* tab, you could see that the bearer token (access\_token) retrieved from previous step would be passed to Copy data activity as **Authorization** under Additional headers. Confirm settings for following properties before starting a pipeline run.

| **Property** | **Description** |
| --- | --- |
| Request method | The HTTP method. Allowed values are **Get** (default) and **Post**. |
| Additional headers | Additional HTTP request headers. |

1. 
2. Select **Debug**, enter the **Parameters**, and then select **Finish**. 
3. When the pipeline run completes successfully, you would see the result similar to the following example: 
4. Click the "Output" icon of WebActivity in **Actions** column, you would see the access\_token returned by the service.



1. Click the "Input" icon of CopyActivity in **Actions** column, you would see the access\_token retrieved by WebActivity is passed to CopyActivity for authentication.

