# Copy from Profisee REST API to JSON Format

This article describes a solution template that you can use to copy records from Profisee REST API to Azure Data Lake Storage Gen2 storage, in JSON format.

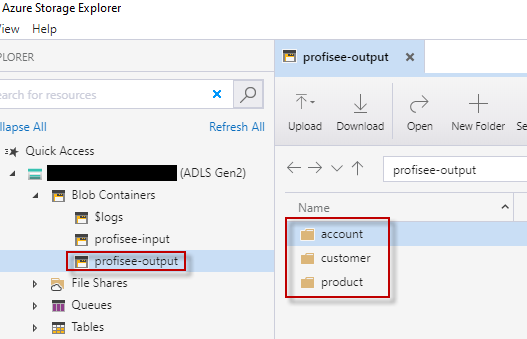
## About this solution template

This template retrieves records from Profisee REST API. It then copies the records, in JSON format, to a file in an output container. The template is designed to work with a folder structure consisting of folders named for each entity within the output container. Create a folder for each entity you wish to integrate with. JSON files for an entity will get created to the profisee-output\<entity> folder.

When the pipeline created by the template is run, it will create a folder for the entity, if it doesn’t exist, and copy the file to that folder. The file name is composed of the entity name and date/time in UTC with the .json extension.

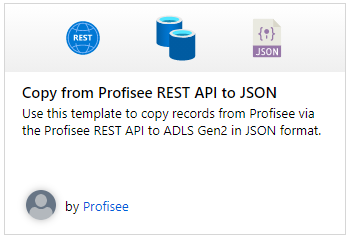
For example:

* profisee-output
  + account
  + customer
  + product

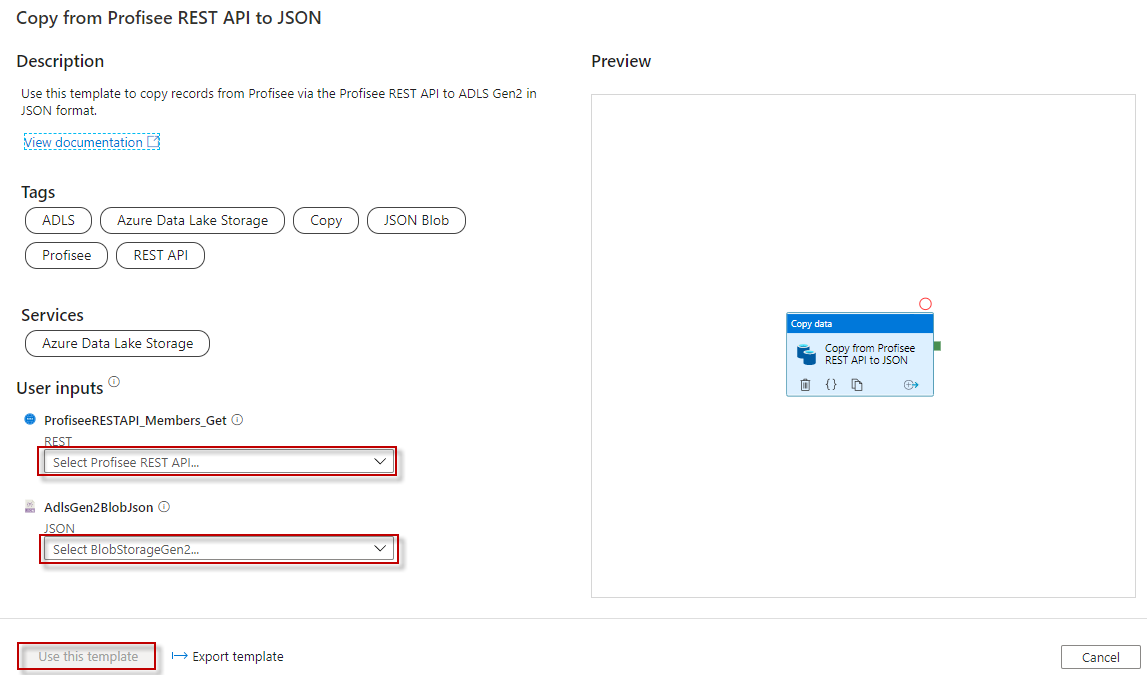


## How to use this solution template

1. Go to the **Copy from Profisee REST API to JSON** template.



1. Create a **New** or use an existing connection to the source Profisee REST API.



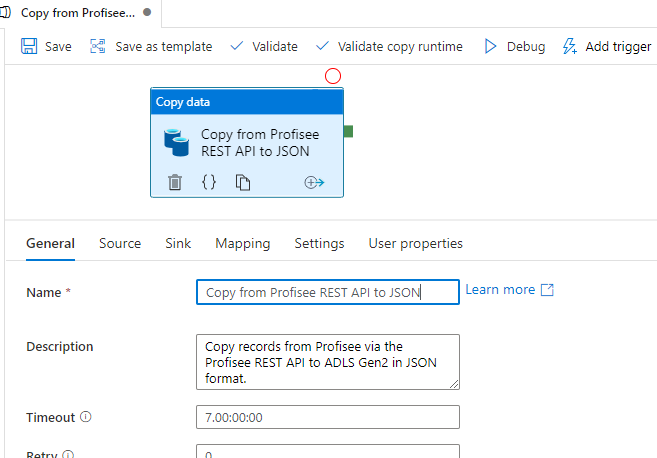
* 1. Follow these steps if you need to create a new REST linked service.
  2. Select “+ New" from the **REST** dropdown list.



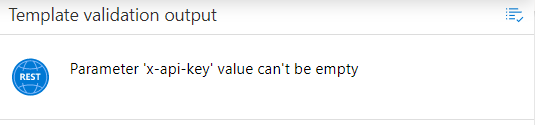
* 1. Enter the following information for the REST linked service.
     1. Name: Enter a unique name within your ADF.
     2. Description: Enter an optional description.
     3. Integration runtime: You can select the auto resolve option or create a custom integration runtime. Some linked services that ADF integrates with requires the ADF integration runtime be in the same region as the service. In this case you will need to create a custom integration runtime in the same region as that linked service.
     4. Base URL: enter the base URL to your deployed Profisee REST API.
     5. Authentication type: select **Anonymous**



1. Create a **New** or use an existing connection to the ADLS Gen2 sink data store that you are copying data to.
2. Select **Use this template**.
3. You will see a pipeline created as shown in the following example:



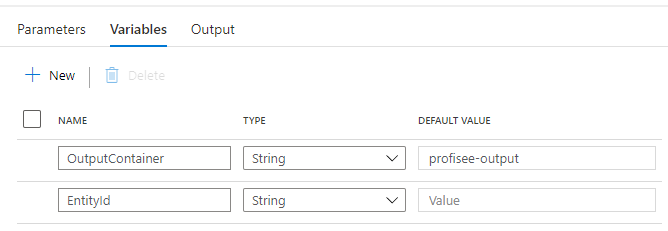
You should also see the following template validation output. We will correct that below.



## Pipeline

### Variables

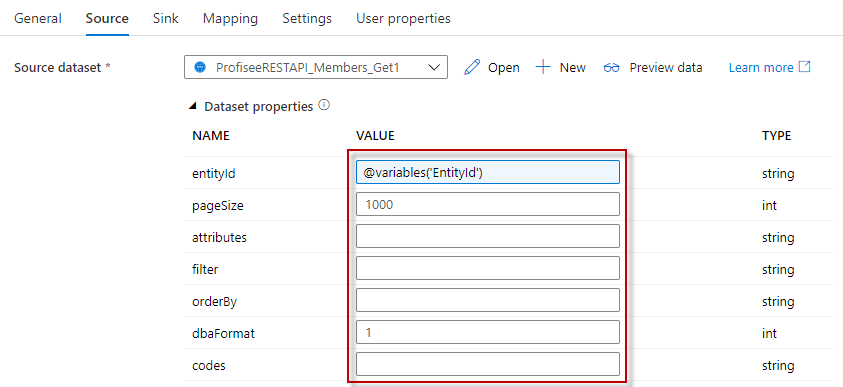
* 1. **OutputContainer:** The output container where you are copying the file to. It defaults to “profisee-output”. You can update to another name based on your environment.
  2. **EntityId:** The entity you are copying records for. Note, the entityId can be either the entity’s Name, UID, or InternalId value.



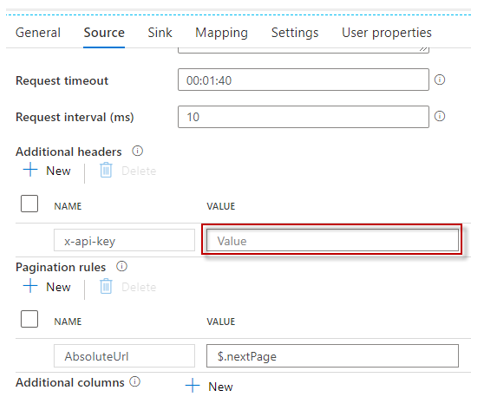
## Copy Activity

### Source

1. Dataset properties:
2. **entityId** - Uses the EntityId variable value.
3. **pageSize** - The page size to get.  Defaults to 1000 if not supplied.
4. **filter** - A filter to restrict the records returned.
   1. [<attribute name>] <operator> <value>.
      * Example: [Color] eq ‘BLU’.
   2. The filter can include multi-level attributes (MLAs).
      * Example: [ProductSubCategory]/[ProductCategory] eq '1'.
   3. You can group attributes together using parenthesis and ANDs and ORs.
5. **attributes** - A comma separated list of entity attribute names to return.  The list can include multi-level attributes (MLAs). If blank, all attributes are returned. Note: the attribute list determines the result properties you will see in the **Mapping** tab.
   1. MLAs are supported, using the ‘/’ to separate each part of the MLA path
   2. Example: [Color],[Class],[ProductSubCategory],[SellStartDate],[SellEndDate],[Weight],[ProductSubCategory]/[ProductCategory]/[ProductGroup]
6. **orderBy** - A comma separated list of entity attribute names and direction to order the response
   1. [<attribute name>] or [<attribute name>] asc - sorts attribute in ascending order
   2. [<attribute name>] desc - sorts attribute in descending order
   3. Example: [ProductSubCategory], [SellStartDate] desc
7. **dbaFormat** - The domain-based attribute (DBA) format to return. Provides an option to indicate how to return the DBA's Code and Name.  Note: a DBA is an attribute that points to, or references, another entity, called a domain entity.
   1. Code only (default) - Only return the code value.
      * Example:
        + "Source System": "SF",
   2. Code and Name simple properties.  The name property is returned as DBA.Name.
      * Example:
        + "Source System": "SF",
        + "Source System.Name": "Salesforce",
8. **recordCodes** – A comma separated list of record codes to restrict the records returned.
9. You can find more information on these parameters on the Profisee REST API Swagger page. You can find it at https://<host name>/Profisee/rest.



1. **x-api-key:** The Profisee API key, which is the Client Id for the user account you are using to connect to the Profisee API. The Client Id can be found in the FastApp Studio Accounts screen, within the Accounts and Teams admin area. This is entered in the Source tab.



### Sink

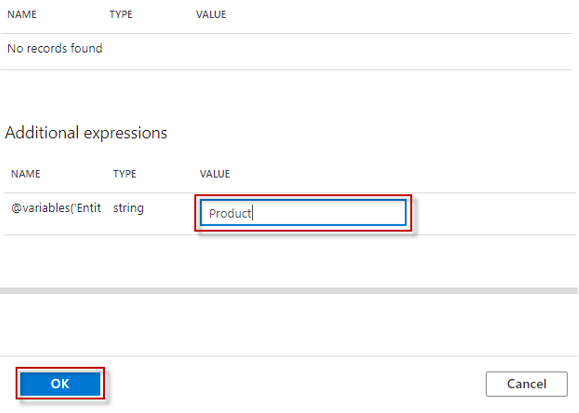
1. Dataset properties
   1. FolderName – A concatenation of the OutputContainer and the EntityId.
   2. FileName – A concatenation of the EntityId and a timestamp.

### 

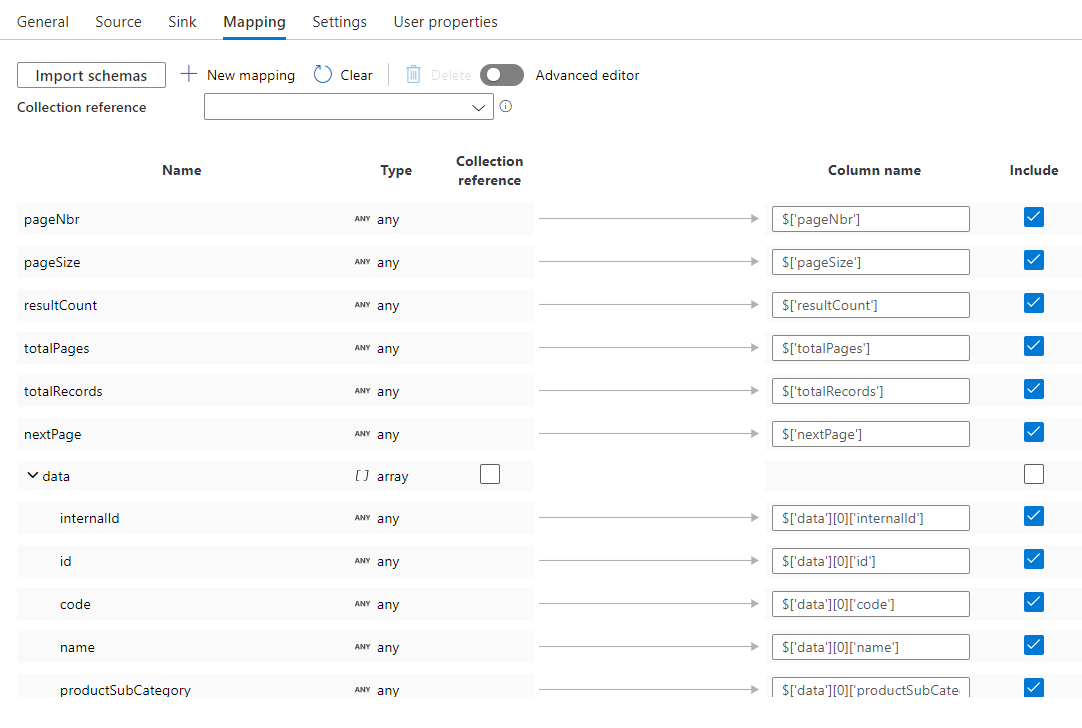
### Mapping

1. Select **Mapping** tab to map the records result properties to the corresponding JSON column.

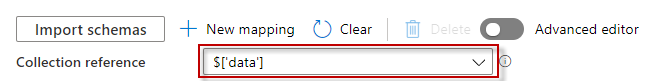
First click the **Import Schemas** button. You will be prompted to confirm the value of the pipeline variable for the EntityId. Click **OK**.



After a couple of seconds, you will see a list of mapping fields listed, as shown in the following example.



Next, select **data** from the **Collection reference** drop down list. The **data** property is the array of records.



Unselect the Include checkboxes for the pageNbr, pageSize, resultCount, totalPages, totalRecords, and nextPage properties as we do not want to copy them to the file.



After selecting the data collection reference, you need to correct the Column names for each property you want to copy by removing the **[‘data’][0]** part of the column name.

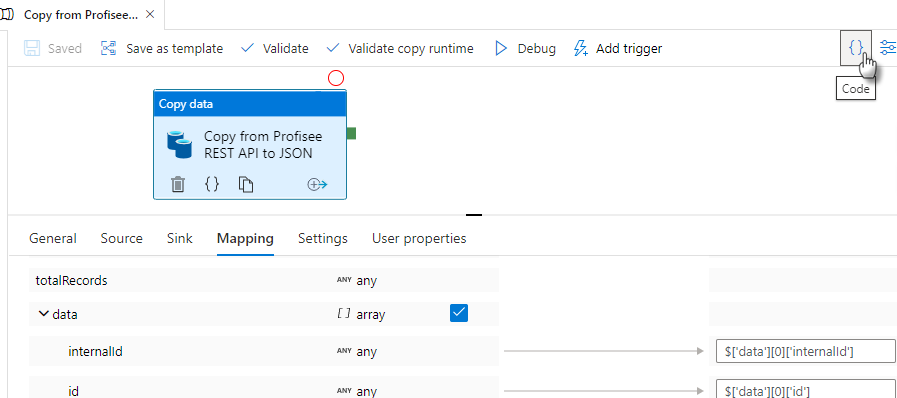
From this



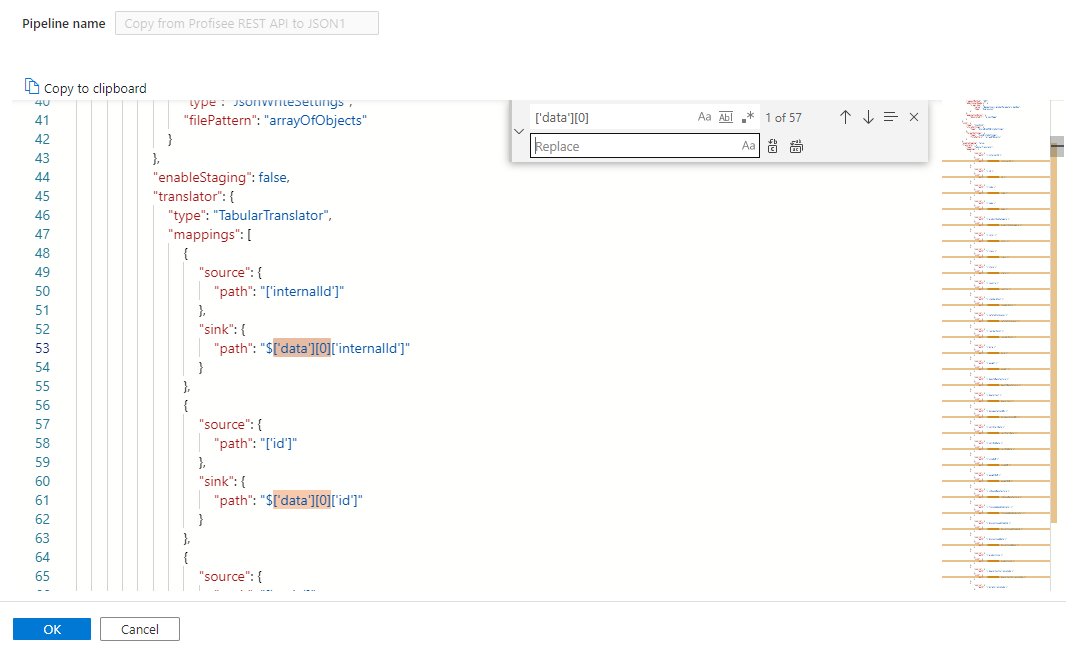
To this



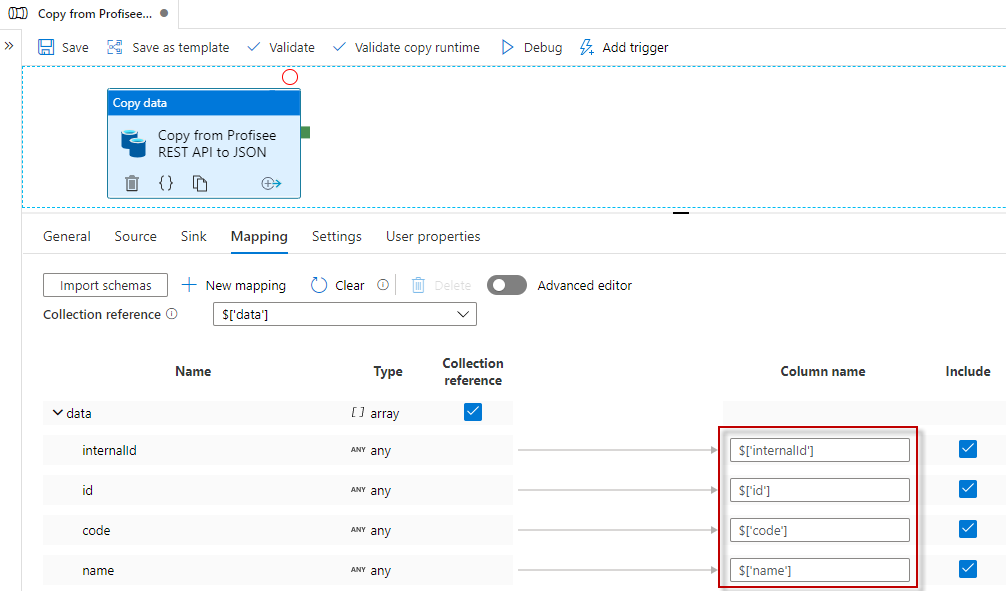
**Quick Tip:** Depending on the number of fields, correcting the column name may take several minutes. Click the { } (Code) icon in the right side of the pipeline toolbar.



Scroll down to the mappings section. Then select **[‘data’][0]** and press Ctrl+H to replace this string with an empty string. Press the Replace all button. Once done replacing, press the OK button.



Go back into the Copy data Mapping tab and now your Column names should all be updated.



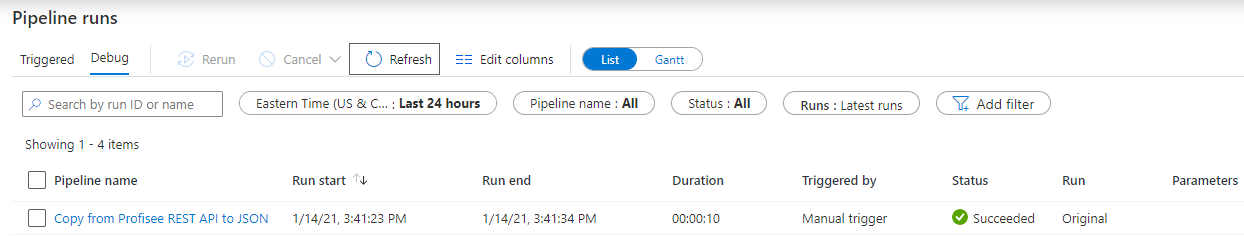
## Publish

Once you are finished with all your changes, click Publish All.

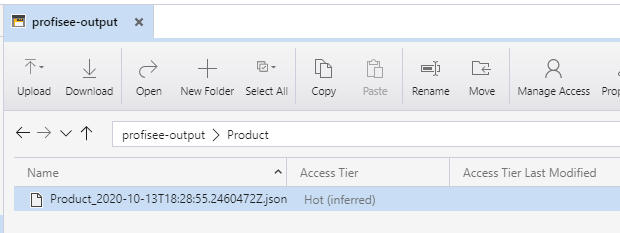


## Triggering

1. To run the pipeline now, select **Add Trigger** and select **Trigger now**. Press **OK** at the Pipeline run prompt.
2. Select **Monitor** tab in the left navigation panel and wait for about 20 seconds. Click **Refresh** to get the updated run status.
3. When the pipeline run completes successfully, you would see results like the following example:



1. You should also see the output file in the Container and Directory you entered.



## Next steps

* [Introduction to Azure Data Factory](https://github.com/MicrosoftDocs/azure-docs/blob/master/articles/data-factory/introduction.md)