

Replicate in 8 steps

Replicate in 8 steps

1. Trigger the flow
2. Authenticate
3. Extract Access token
4. Assign Variables
5. Do Until Loop
6. Execute the Query
7. Parse the results
8. Create/Replicate the data item

-Services and solutions are available and potential options

-Build flows to replicate you data

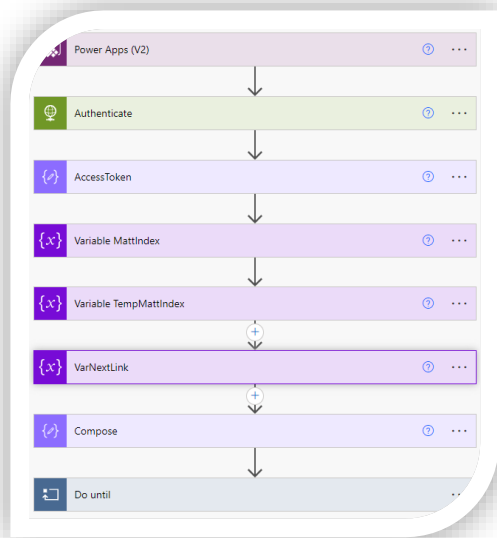
-Trigger from a button

-Trigger from a cloud event

How can you replicate data from a cloud system? A new matter has just been created upstream, and you need to automate insert the row/details more local to your organization for reporting purposes or building custom flows where the data needs to reside closer to home.

Power Automate can accomplish. -Concerned about processing 100s,1000s,10,000s, 100,000s of rows? -PowerAutomate can handle this. -SharePoint can handle this.

Recently, I was engaged for a flow to accommodate such a scenario. To process 100,000 rows did take about two hours to replicate, create the storage item (in SharePoint) and perform additional tasks. But batches of 5000 rows happens pretty quick. -Moreover, after you have your initial load, processing this many rows moving forward should not be a requirement. Afterall, you are only looking for your delta or net new rows.



Replicate data with a few steps. Query with an HTTP step/operation, and iterate those rows until a condition a met. Also, you could execute the query until a ‘Do Until’ loop is executed.

Sometimes, your data is stored in the cloud but needs to be accessed in other locations. For example, let’s say you’re working with a matter management system that stores data in the cloud, but you need to replicate this data to a SharePoint list or SQL table for reporting or external analysis. When building extensions, apps, or flows, having replicated data in an easily accessible location, like SharePoint or SQL, can help streamline those processes. Additionally, when you want to create custom reports with tools like SSRS or Microsoft Access, replicating that data becomes the ideal solution to centralize the needed information.

Why Replicating Data Is a Great Option (with use cases)

1. **Custom Reporting:** Replicating data into SharePoint or SQL allows the use of advanced reporting tools such as SSRS, which may require a local dataset to function efficiently.
2. **Cross-System Integration:** Many times, external applications or services need access to your data. By replicating the data to a SQL table, third-party integrations can read this data without directly querying your cloud system, which reduces system load.
3. **Data Archiving:** Cloud data can change frequently, and sometimes you need to retain a snapshot of data at a certain time. Replicating this data into a local or on-premise system (like SharePoint or SQL) ensures that you have a copy to reference at a later point.

Summary of Steps

Here's how you can build your Power Automate flow to replicate data. Get ready, because it's going to be fast-paced and effective! You'll create a seamless connection between your cloud system and the data repository of your choice, looping through records with precision until the work is done.

In this setup, we'll be querying data, looping through batches, and inserting them into your chosen storage. Let's dive in and get that replication flow up and running in no time!

Step-by-Step Guide for the Power Automate Flow:

1. **Trigger the Flow:** Start by triggering the flow from a button or Power App.
2. **Authenticate:** Use an API connector or HTTP request to authenticate to the system. Make sure you store the authentication token securely.
3. **Save Values:** Initialize and save the values needed, such as counters and the starting index of the data (e.g., the starting matter index).
4. **First Query:** Make your first query using the HTTP action to retrieve the first batch of records based on the @odata.nextlink. The response will contain the next link for fetching additional records.
5. **Create a Do Until Loop:**
 - Inside this loop, execute the query to retrieve batches of matters starting with the saved matter index.
 - Store the next link (OData.NextLink) for each iteration.
6. **Increment:** Continue querying and incrementing the matter index or checking the next link value until the end is reached (OData.NextLink is null).
7. **Parse Data:** Use a schema to extract the values needed from each record.
 - Let the flow fail once during setup, then copy and paste the schema generated from Power Automate for easy parsing.
8. **Save Data:** Loop through each record and store the desired fields in your preferred data storage, such as a SharePoint list or SQL table.

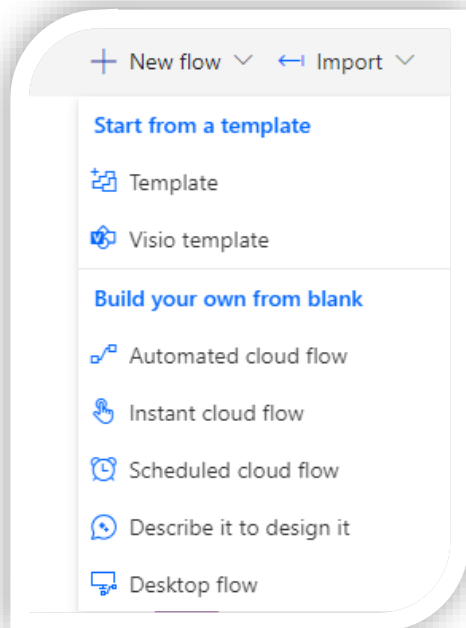
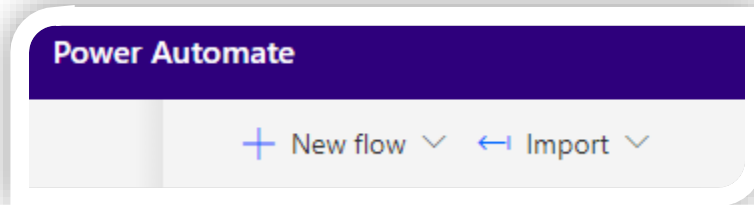
How to Create a SharePoint Hidden List:

1. Navigate to your SharePoint site.
2. Click on **Site Contents > New > List**.
3. Name your list, then click on **List Settings**.
4. Under **Advanced Settings**, set the list to **Hidden**. This will ensure it doesn't appear in the site navigation but is still accessible to your flows.

Begin the workshop





From make.powerautomate.com

Create a new flow



Select “manual trigger a flow” if you want to run from the flow itself. Alternatively, if you are building a “Power App” select the Power Apps option.

Choose how to trigger this flow *

-  Manually trigger a flow
Flow button for mobile 
-  When Power Apps calls a flow (V2)
Power Apps 



1. Trigger the Flow

Create step to run the flow.

Click the PLUS on the canvas to add new steps.

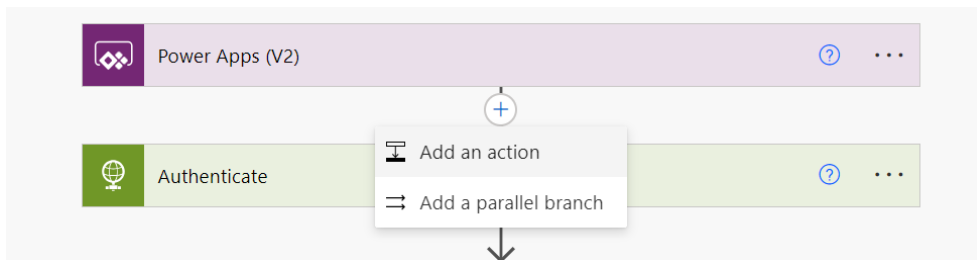


2. Authenticate

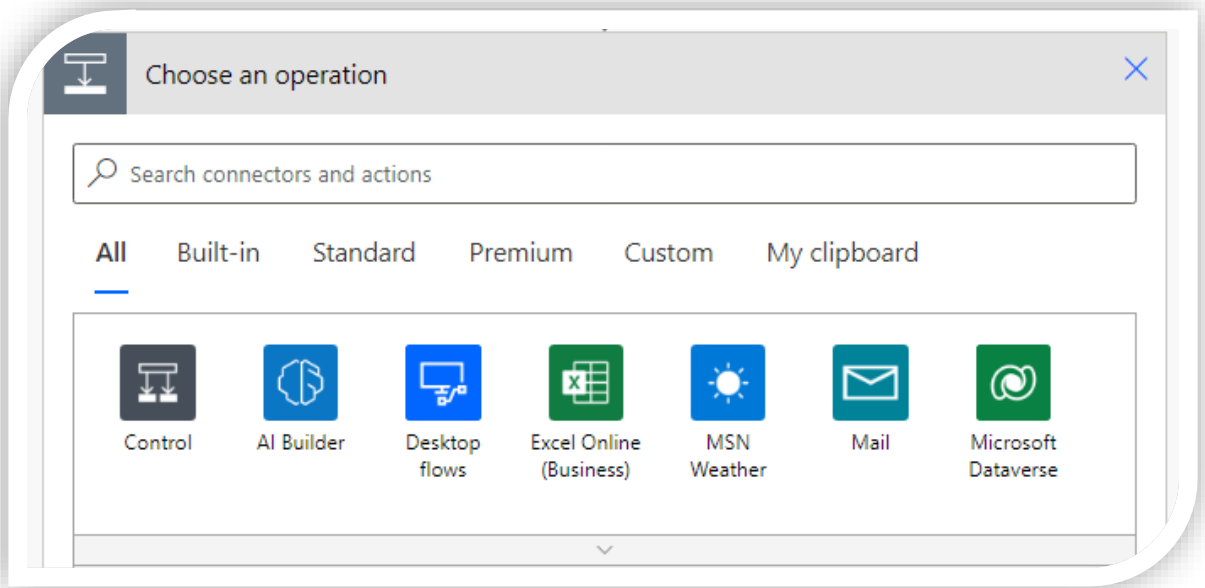
Best Practice: Save these items in a SharePoint hidden list or environment variable.

Create a step with the "HTTP" action request

Click on the "+" and type "HTTP". Select the operation from the list.



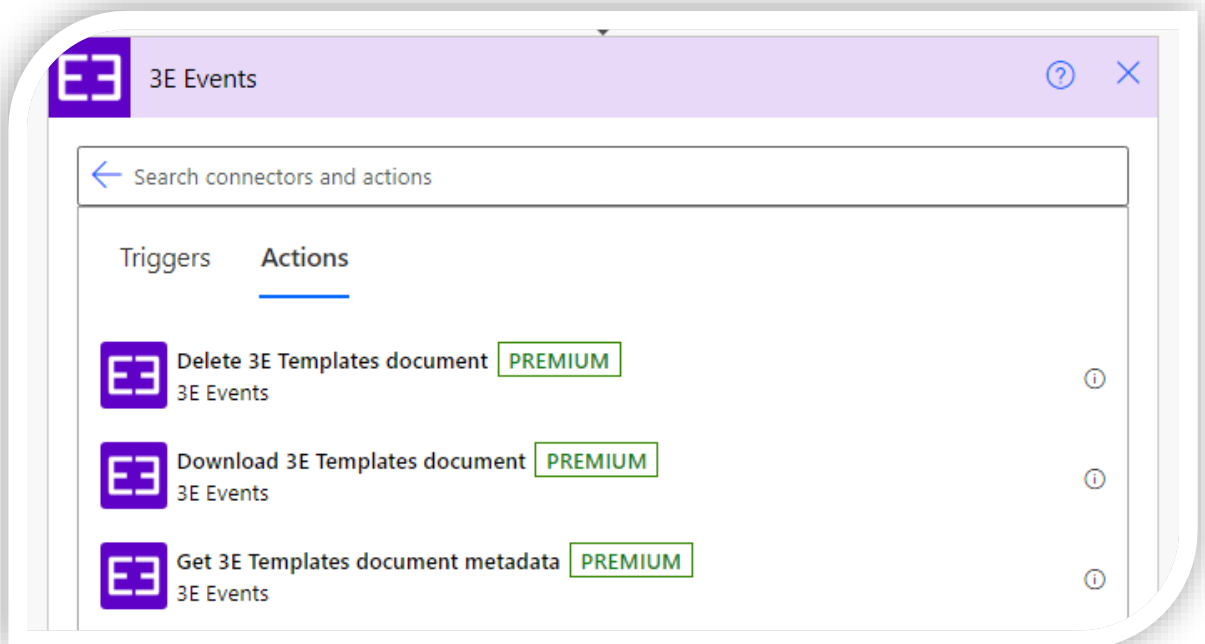
Choose from the source to integrate. This is your list of Powe Automate connectors which provides "integration" to many systems.

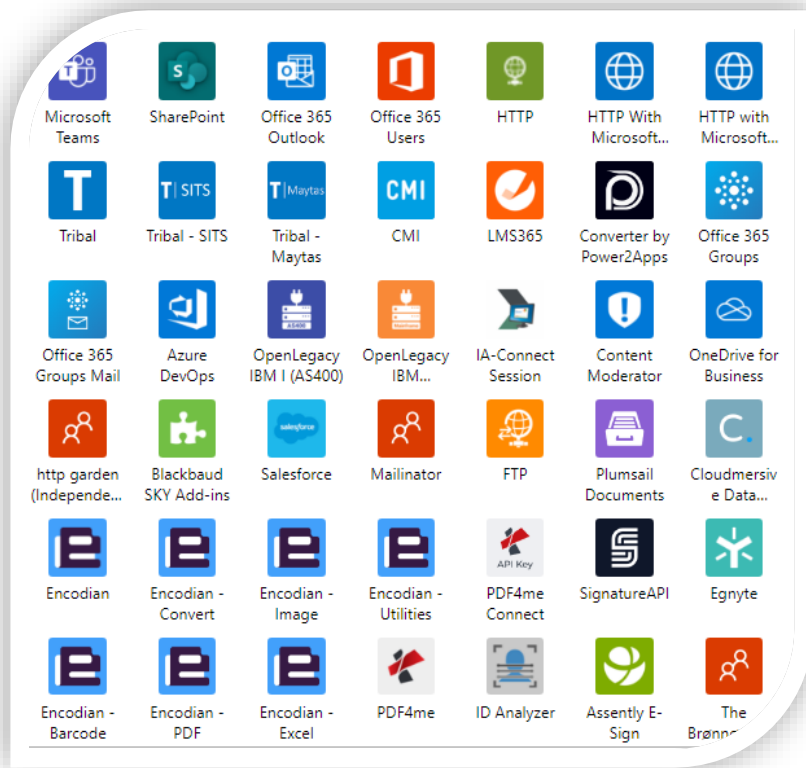


NOTE: here is a quick short list of connectors you might implement.

-Actions or events you might take with Elite 3E

‘Packaged webhooks/APIs you can implement





iManage



iManage
Work



iManage
Tracker



iManage
Work for...



iManage
Insight Plus



Get permissions **PREMIUM**
iManage Work



Move document **PREMIUM**
iManage Work



Update current or create new document version **PREMIUM**
iManage Work








Get document profile **PREMIUM**
iManage Work



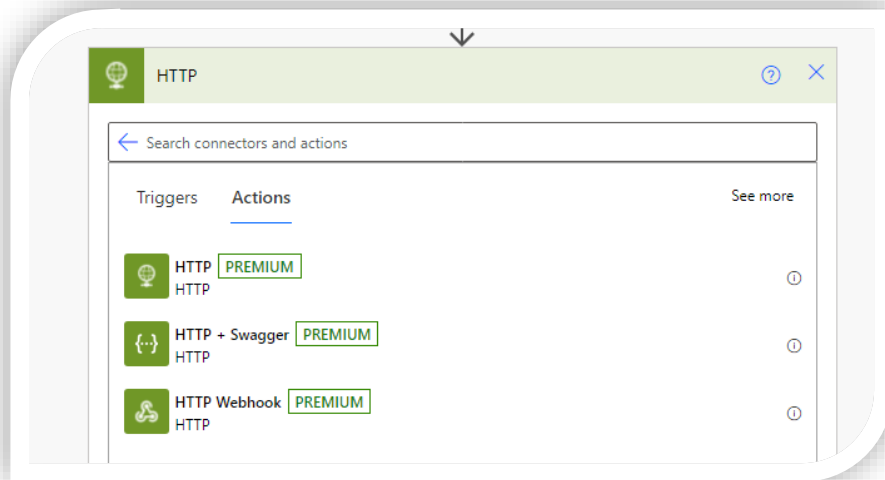
Get document versions **PREMIUM**
iManage Work



Get user details **PREMIUM**
iManage Work

-  Get workspace profile **PREMIUM**
iManage Work
-  Get core extended metadata properties of a document **PREMIUM**
iManage Work
-  Add document reference **PREMIUM**
iManage Work
-  Copy document **PREMIUM**
iManage Work
-  Update permissions **PREMIUM**
iManage Work

For your flow, select the standard “HTTP” request.



Select “HTTP” from this option.

Configure the step as follows:

Prerequisites: before you can build this flow, precept information will be required.

1. 3E Instance ID
2. API permissions
3. Credentials

The screenshot shows a configuration window titled "Authenticate" with the following fields:

- * Method:** POST
- * URI:** https://login.microsoftonline.com/<id>/oauth2/v2.0/token
- Headers:**

Content-Type	application/x-www-form-urlencoded	✕
X-3E-InstanceID	<instanceid>	✕
Enter key	Enter value	
- Queries:**

Enter key	Enter value	
-----------	-------------	--
- Body:**

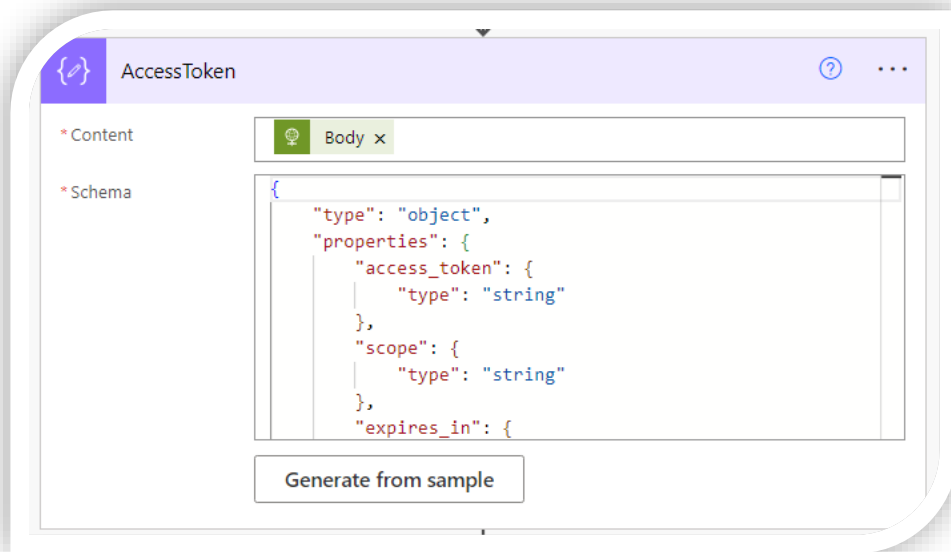
```
grant_type=client_credentials
&client_id=<clientID>
&client_secret=<secretvalue>
&Scope=<systemscopeurl>
```
- Cookie:** Enter HTTP cookie

At the bottom, there is a link "Show advanced options" with a downward arrow.

3. Extract the token

Add a data operation step for “Parse JSON” –(not this JASON)

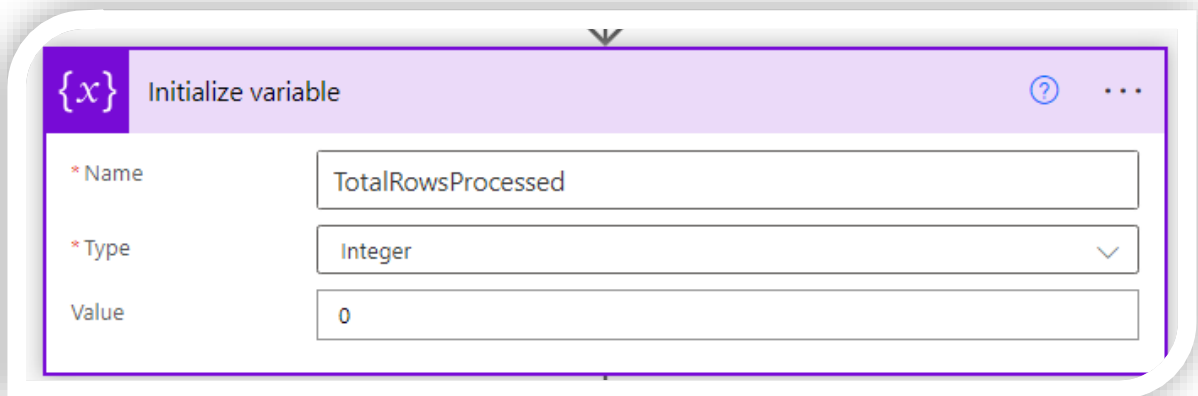
Parse the returned results and extract the token. You will need the token to execute the query in the next step. From the Body of the response, you need to extract the token.



4. Initialize Starting Point

Initialize and save values for counters and the starting matter index.

Using the step create a variable.



@Odata.NextLink

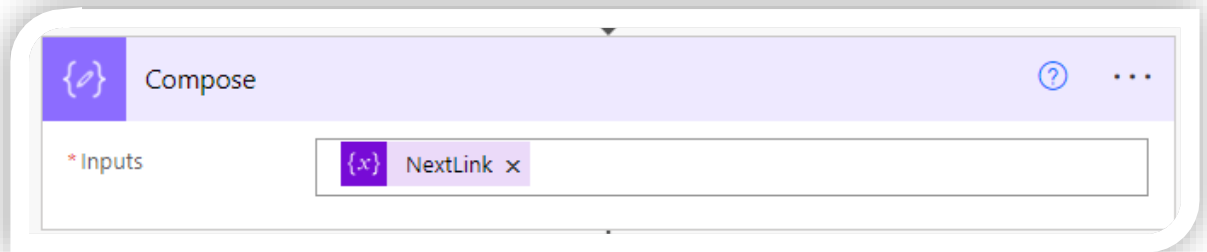
Create a variable to save the OData.Next link.

WIN FOR THE CLOUD: Returned instructions for finding the next batch of records
 data/CostCard?\$skip=1000&\$top=1000&\$count=true&\$orderby=costindex%20asc&\$filter=IsHar
 rdCost%20eq%20true&\$select=*,CostIndex,Currency,Matter,CostType

Compose

Save the NextOData link.

Create a variable using the data operation, “Compose” to store the JSON variable.



5. Do Until Condition is satisfied

Inside the “Do Until” add an apply each loop control.

Do this until the matter index reaches your threshold limit. (5000).

Steps for Do Until Conditions:

Using MattIndex Greater than a Variable:

1. Inside the **Do Until** loop, set the condition as:
MattIndex <= {your upper limit variable}
2. For each iteration, update the MattIndex by adding the batch size (e.g., increment by 5000).
3. Continue processing until MattIndex exceeds the upper limit.

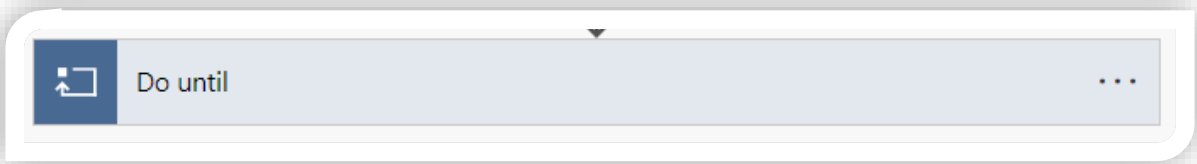
Using OData.NextLink is Null:

1. Inside the **Do Until** loop, set the condition as:
OData.NextLink != null
2. Query the data and store the OData.NextLink in a variable.
3. For each iteration, execute the query using the saved next link and update the variable with the new OData.NextLink.
4. Continue until the next link is null, indicating the end of the data.

Inside the Do Until loop:

Execute a query to find the batch of matters greater than your starting matter index.

Create a step with the Do Until action.



6. Execute the query

The screenshot shows the 'EliteQuery' tool interface with the following configuration:

- Method:** GET
- URI:** <baseserviceurl>/Matter?
- Headers:**
 - Authorization: Bearer access_token
 - X-3E-InstanceID: <yourinstanceid>
- Queries:**
 - \$select:** MattIndex,Number,DisplayName,Client,MattStatus,OpenDate,OpenTkpr,MatterAdditionalInformations(\$Select=Code,FieldValue;\$filter=Code eq 451),MattDates(\$Select=EffStart,PracticeGroup,Department,Section,BillTkpr,RspTkpr,SpvTkpr)
 - \$expand:** Client1(\$Select=Number,DisplayName)
 - \$filter:** MattIndex gt mattindex
 - \$orderby:** MattIndex asc
 - \$top:** 5000
- Body:** Enter request content
- Cookie:** Enter HTTP cookie

At the bottom, there is a 'Show advanced options' dropdown menu.

7. Parse the Results



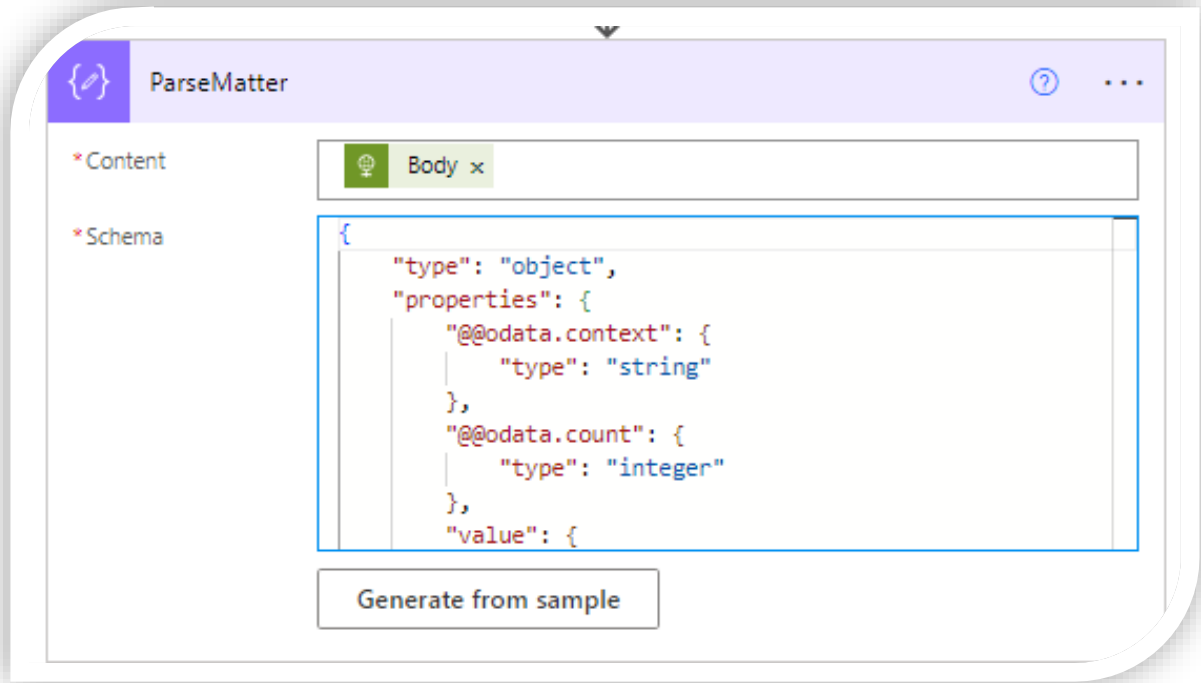
Format a schema to extract the field values.

Sample of a matter schema:

```
{  
  
  "type": "object",  
  "properties": {  
    "@odata.context": {  
      "type": "string"  
    },  
    "@odata.count": {  
      "type": "integer"  
    },  
    "value": {  
      "type": "array",  
      "items": {  
        "type": "object",  
        "properties": {  
          "MattIndex": {  
            "type": "integer"  
          },  
          "Number": {  
            "type": "string"  
          },  
          "DisplayName": {  
            "type": "string"  
          },  
          "Client": {  
            "type": "integer"  
          },  
          "MattDates": {  
            "type": "array",  
            "items": {  
              "type": "object",  
              "properties": {  
                "BillTkpr": {  
                  "type": "integer"  
                }  
              }  
            }  
          }  
        }  
      }  
    }  
  }  
}
```

```
"MatterAdditionalInformations": {
  "type": "array",
  "items": {
    "type": "object",
    "properties": {
      "MatterAdditionalInformationID": {
        "type": "string"
      },
      "Code": {
        "type": "integer"
      },
      "FieldName": {
        "type": "integer"
      },
      "FieldValue": {
        "type": "string"
      },
      "MatterId": {
        "type": "integer"
      }
    }
  }
},
"Client1": {
  "type": "object",
  "properties": {
    "Number": {
      "type": "string"
    },
    "DisplayName": {
      "type": "string"
    }
  }
},
"required": [
  "MattIndex",
  "Number",
  "DisplayName",
  "Client"
]
},
"@odata.nextLink": {
  "type": "string"
}
}
```

Provide a schema: Power Automate will generate this in the HTTP query step. Let your flow fail intentionally so you can copy and paste the schema—a nice hack.

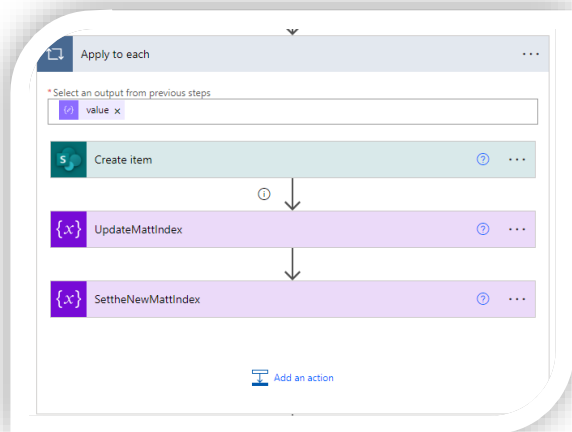


8. Replicate the data

Inside the “Do Until” loop create an apply each.

-New Step, create an “Apply to Each” step.

Inside the apply each loop (which is iterating all of the rows). Loop through all of the matters returned from the query. Example provided batches 5000 rows. Start with your initial index. You will increment the matter index number. The query is dynamic to start with the matter index and increase the threshold.



Inside the apply each loop (which is iterating all of the rows). Loop through all of the matters returned from the query. Example provided batches 5000 rows. Start with your initial index. You will increment the matter index number. The query is dynamic to start with the matter index and increase the threshold.

Add a step “SharePoint, create item”



Instead of creating items in a SharePoint list, insert rows into a SQL table, data warehouse, or preferred data storage.

```
plaintext Copy code
Title: @{items('Apply_to_Each')['DisplayName']}
MatterIndex: @{items('Apply_to_Each')['MattIndex']}
ClientNumber: @{items('Apply_to_Each')['Client1']['Number']}
ClientName: @{items('Apply_to_Each')['Client1']['DisplayName']}
```

Extract the desired fields using syntax as follows:

```
Title: @{items('Apply_to_Each')['DisplayName']}
MatterIndex : @{items('Apply_to_Each')['MattIndex']}
ClientNumber : @{items('Apply_to_Each')['Client1']['Number']}
ClientName: @{items('Apply_to_Each')['Client1']['DisplayName']}
```