**3-2– Batch Macros** 

**Both the Iterative and Batch Macro runs repeatedly based on a certain criterion. Iterative Macro runs indefinitely and only stops once it meets a condition while Batch Macro has a definite number of runs which depends on the number of items. So, what exactly is a batch macro and how can it improve our alteryx workflows? A Batch macro runs multiple times in a workflow and creates an output after each run. The macro runs once for each record (or a selected group of records) in the data. Inputs can be configured to be used in each run of the workflow or only in specific runs.**

To make it easier to understand, a batch macro can be likened to a delivery service truck. It is impossible to deliver all the packages in a big sorting warehouse if you only have a single truck. In order to effectively send all packages to each address, you need to **batch** them by location, size and expected time for delivery. By batching the packages, you can create a clear route for addresses that are within a specific area, and schedule which packages might take more time to deliver and are mostly out of the way.

Let’s create a batch macro that will generate a pdf file for each supplier. Each page of the pdf will contain a different product category. The products names will be the header for each table column and its respective order amount will be the value. Our original data sources come from 3 sheets; Product, Category, and Supplier. Product was joined to Category using CategoryID, then its inner join output was joined to the Supplier using SupplierID. Blending the 3 sources into 1 data stream makes it easier for us to create a batch macro since we do not need to add several input anchors. We will create a macro that takes the data after unnecessary fields are removed by the select tool. This leaves us with 4 columns to work with; Product Name, Orders, Category Name and Supplier. Copy to clipboard the first 6 records with Headers then paste them into a new workflow. Once pasted, the data should be automatically turned to a Text Input. Add a select tool to correct the data types and size. Set all of the string columns to “V\_String'' size 255 except for Orders. Orders should be set to Int64. This will serve as the template or test data for our macro.

Click right on the text input and convert it to a macro input. Once converted, set the Input’s anchor abbreviation to “I” and retain the default input name of “Input1”. Next step is to change the type of macro. Click on any blank part of the canvas then navigate to the workflow tab of the configurations window. Under type, change the macro to a “Batch Macro”. Going back to our canvas, the next tool to add is a filter. Drag a filter from the preparation tool palette and connect it to the output anchor of the select tool. Since we are going to process the data by individual suppliers, we need to filter a specific test supplier’s name into the filter tool. Set the configuration of the tool to a basic filter with the condition “Supplier equals ABC”. The ABC supplier was included as the first 3 rows of our test data. Make sure to match your filters with that of your test data so you can easily check how it flows in your macro. We will also use supplier as the grouping or the batch iteration number. So, if there are 29 suppliers on our data, the macro will run 29 times, 1 for each unique supplier. In order to set this certain field as an iteration control field, we need to add a Control Parameter.

**A Control Parameter is the input for each iteration of a Batch Macro. The Control Parameter input appears as an upside-down question mark icon below its input arrow on the macro tool icon. For each record coming into the Control Parameter input, the entire macro is reconfigured and runs from beginning to end. Like other interface tools that take in user input, this tool will always require an associated action tool.**

From the Interface tool’s palette, drag a control parameter to the canvas. Connect its Q anchor to the lightning bolt anchor of the filter to automatically create an action tool. Then, configure the control parameter. This tool only has a single configuration which is to set the text or question to be displayed. Type in “Control Parameter (map to Supplier)” Next, configure the action tool. Set the action type to “Update Value”. Once done, set the value or attribute to update. Double click “Simple” to show its filter attributes. Then click the + button on operands to show its tree attributes as well. From this list choose the line for Operand Value= “ABC”. After selecting the attribute, tick the checkbox on the bottom part to “Replace a specific string” and type the test supplier name “ABC”. This process is applied in order to change which supplier is being processed while running a different iteration of the workflow. For example, the first run will be products ordered from “ABC”, while the 2nd run will be products ordered from “Asin Fashion Ltd Co” and so on.

Now that we have the batch macro’s essential tools in place, we can now add additional tools for the preparation and creation of each pdf report. From the transform palette, drag a cross tab and connect it to the output True Anchor of the filter tool. Set the column header to “ProductName”, and its value as “Orders”. Since we want each page to be separate by category, under group data, tick the checkbox for Category Name and Supplier. This is done in order to include the 2 fields on the output. Finally, set the method for aggregating values. Select Sum. Once all settings are done, run the workflow to see a preview of our data. The crosstab tool has minimized our rows to a single line since we only have 1 category for the ABC supplier. Each product was also assigned with their own column. Now we can proceed with formatting it to a report table. From the Reporting palette, drag a Table tool into the workflow. Set the table to Group by “Category Name” and “Supplier”. We had to group by category since we want it separated by page. For the table columns, uncheck the Category Name and Supplier then retain the product columns. Also make sure that the Dynamic or Unknown Fields is checked so that products from other suppliers will not be hidden. Next, configure how the table will look like. Click the default table settings and set the Data Font to Calibri, with size 12. Then on the headers tab, set the Header Font to Calibri and enable the style for “Bold”. Set the size of the header to 14. For the colors, set the text color to White and its Background Color to Teal (or RGB 64,128,128). This completes the configurations for the table. Aside from the table, it would be better to have the Category Name displayed on each page as Title or Header. From the Reporting palette, drag a Report Text tool and place it after the Table tool. Set the text mode to “Attach text to existing field”. Set the field name to “Table” – which contains the field object created by the table tool then position it to “Place text above existing object”.  
As for the text data, click the font button first, and change it to Calibri BOLD 16. Click Apply once the font style has been set. Next, click the center alignment. After that, click the available fields and select “Category Name” to insert it as text.

Before generating our output to a pdf, add a browse tool after the report text tool then run the workflow. Under the Report tab of the browse tool, we can now see the table generated from the configurations we’ve applied earlier. A category name header is also attached on top of the table. Finally, add a render tool and connect it to the report text tool. Set the Output Mode to “Choose a Specific Output File”. Then click the File browse icon and navigate to your desired output path. For this example, we are also going to name the default pdf name as “-orders.pdf” since we are going to add the supplier’s name as a prefix. Enable the function “Group Data into Separate Reports”. Then set the grouping field to “Supplier” and modify the filename by “Prepending Group to Filename”. Now to separate the category by page, under the report data section, set the data field to Table, then add a separator. On its dropdown choose to “Insert Section Breaks Between Records”.

To finish our batch macro, open the interface designer and apply a custom icon. Since we will not generate an output data stream, we do not need to configure the Output Mode. Save this macro to your macro folder. We will name this example as “3-2-pdfcreator.yxmc”. To use the macro, go back to the original alteryx workflow with a complete dataset and insert the created macro. Notice that it has 2 input anchors. The inverted question mark is for the control parameter which sets the batches or the grouping while the I anchor is for the data stream. To make the supplier list unique, add a summarize tool after the select tool and set it to group by supplier. Connect the output of the summarize tool to the inverted question mark anchor. Next, connect the select tools output to the I data anchor. The macro’s configuration has 2 tabs, Group BY and Questions. We already grouped the supplier field so there is no need to add another grouping. Set the control by group field to None. For the 2nd tab, we need to map the control parameter. Select “Supplier” in its dropdown. Once done, run the workflow.

On the pdf output folder, we can see the 29 pdf files that the macro created. If we open the “Austerlich-orders.pdf” we can now see 5 separate pages, 1 for each category.