PharmaSUG 2025 - Paper RW-374

The Many Ways to Build Cohorts to Effectively Generate Real World Evidence and Bring Drugs to Patients Faster

Sherrine Eid, MPH, Mary Dolegowski, SAS Institute, Inc.

ABSTRACT

The process of defining and building patient cohorts in pharma today involves a mix of automated and manual methods that leverage structured data, unstructured data, and advanced analytics. Each approach carries trade-offs between scalability, accuracy, interpretability, and resource requirements. The importance of these methods lies in their ability to produce reliable real-world evidence that informs clinical practice, supports regulatory decisions, and ultimately improves patient outcomes.

Properly defining patient cohorts affects study validity and reliability as accurate definitions reduce bias and confounding, ensuring that the evidence generated is robust and valid. They enable meaningful comparisons between treatments and help in drawing reliable conclusions about effectiveness and safety.

Regulators and payers increasingly rely on RWE to complement randomized clinical trials (RCTs) in decision-making. There is a plethora of examples illustrating how well-defined cohorts support regulatory submissions, label expansions, and post-marketing surveillance.

Identifying distinct patient subgroups can lead to targeted therapies and more personalized treatment strategies. This enables the exploration of treatment effects in real-world subpopulations that might be underrepresented in RCTs.

Standardized and reproducible cohort definitions streamline the research process, making studies more efficient and scalable and facilitates data sharing and collaboration across institutions, which is crucial for large-scale observational studies.

This session will explore the many options researchers have at their disposal to efficiently and accurately define cohorts and accelerate evidence generation in support of drug development and integrated evidence packages.

INTRODUCTION

The process of defining and building patient cohorts in pharma today involves a mix of automated and manual methods that leverage structured data, unstructured data, and advanced analytics. Each approach carries trade-offs between scalability, accuracy, interpretability, and resource requirements. The importance of these methods lies in their ability to produce reliable real-world evidence that informs clinical practice, supports regulatory decisions, and ultimately improves patient outcomes.

Properly defining patient cohorts affects study validity and reliability as accurate definitions reduce bias and confounding, ensuring that the evidence generated is robust and valid. They enable meaningful comparisons between treatments and help in drawing reliable conclusions about effectiveness and safety.

Regulators and payers increasingly rely on RWE to complement randomized clinical trials (RCTs) in decision-making. There is a plethora of examples illustrating how well-defined cohorts support regulatory submissions, label expansions, and post-marketing surveillance.

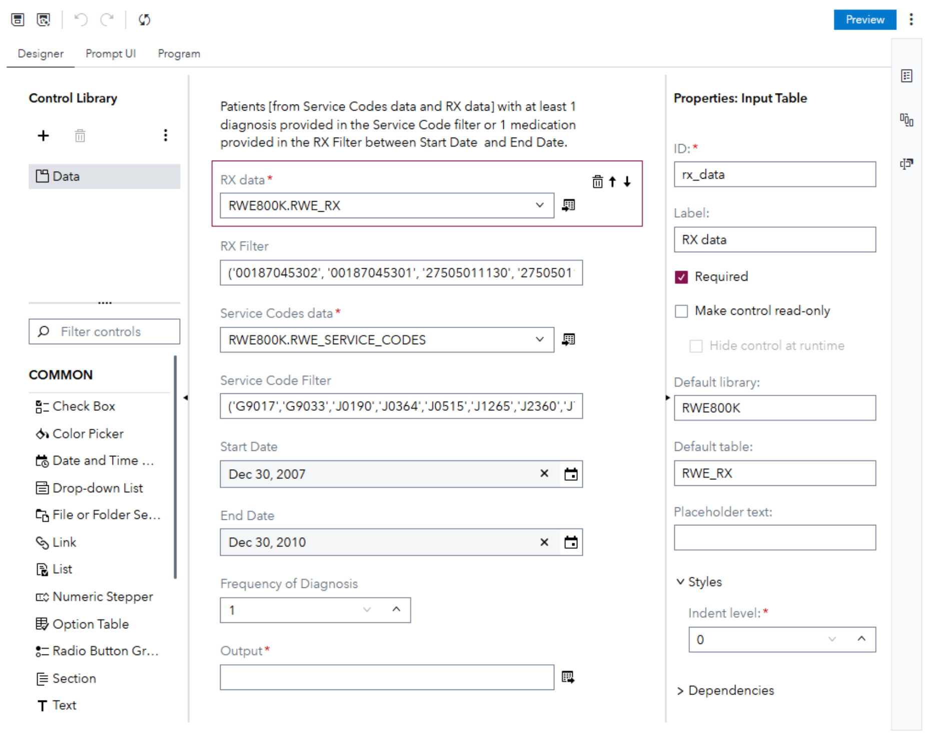
BUILDING CUSTOM STEPS

SAS Viya Custom Steps allow users to create repeatable code chunks that are integrated with a graphical user interface (GUI). This enhances accessibility for downstream users who may not be familiar with the underlying code, while also reducing the likelihood of errors related to data selection and manipulation.

CUSTOM STEP DOMAINS

Within the configurations for a Custom Step, there are three main domains: Designer, Prompt UI, and Program. The Designer tab provides a GUI for building the controls that will be presented to the end user. Controls are added using drag-and-drop functionality, and their specific properties can be configured in the Properties panel, as shown in Display 1.

For example, the control highlighted in Display 1 is an input data control with its ID set to 'rx_data'. This ID will be referenced in the Program tab, where it functions as a macro variable. The designer can configure additional settings using the Properties panel, such as making the field required, setting default values, or establishing more complex behaviors like dependencies. This flexibility allows designers to create user interfaces that range from simple to highly customized, depending on the needs of the end users.



Patients [from Service Codes data and RX data] with at least 1 diagnosis provided in the Service Code filter or 1 medication provided in the RX Filter between Start Date and End Date.

RX data*: RWE800K.RWE_RX

RX Filter: ('00187045302', '00187045301', '27505011130', '2750501

Service Codes data*: RWE800K.RWE_SERVICE_CODES

Service Code Filter: ('G9017','G9033','J0190','J0364','J0515','J1265','J2360','J

Start Date: Dec 30, 2007

End Date: Dec 30, 2010

Frequency of Diagnosis: 1

Output*:

Properties: Input Table

ID:*: rx_data

Label: RX data

☑ Required

☐ Make control read-only

☐ Hide control at runtime

Default library: RWE800K

Default table: RWE_RX

Placeholder text:

Styles

Indent level:*: 0

Dependencies

Display 1: Custom Step – Designer

In Program 1, we see the code associated with the design shown in Display 1. This code includes a collection of macro variables that correspond to various controls defined in the Designer tab. For example, 'select CONSISTENT_MEMBER_ID from &rx_data. where CODE_VALUE in &RX_filter.' references the macro variables &rx_data and &rx_filter. These variables are directly linked to the controls labeled RX Data and RX Filter in Display 1.

This relationship between the Designer and Program tabs makes it easy to adapt existing code for use within a Custom Step. By simply converting user-configurable elements into macro variables, developers can integrate a GUI interface without rewriting the entire program.

```
/* SAS templated code goes here */

proc fedsql sessref = mysession;

      create table and_Filtered_Member_IDs as
      select DISTINCT _RX_Members.CONSISTENT_MEMBER_ID
      from
```

```
              (select CONSISTENT_MEMBER_ID from &rx_data. where CODE_VALUE in &RX_filter.)
              as _RX_Members
      INNER JOIN
              (select CONSISTENT_MEMBER_ID from &code_data. where CODE_VALUE in
&code_filter.)
              as _Code_Members
      ON _RX_Members.CONSISTENT_MEMBER_ID = _Code_Members.CONSISTENT_MEMBER_ID
      ;

      create table Filtered_RX as
      select RX.*
      from &rx_data as RX
      INNER JOIN
      (
              select DISTINCT _RX_Members.CONSISTENT_MEMBER_ID
              from
              (select CONSISTENT_MEMBER_ID from &rx_data. where CODE_VALUE in &RX_filter.)
              as _RX_Members
              INNER JOIN
              (select CONSISTENT_MEMBER_ID from &code_data. where CODE_VALUE in
&code_filter.)
              as _Code_Members
              ON _RX_Members.CONSISTENT_MEMBER_ID = _Code_Members.CONSISTENT_MEMBER_ID
      )
      as F
      ON RX.CONSISTENT_MEMBER_ID = F.CONSISTENT_MEMBER_ID
      ;

      create table Filtered_Service_Code as
      select code.*
      from &code_data as code
      INNER JOIN
      (
              select DISTINCT _RX_Members.CONSISTENT_MEMBER_ID
              from
              (select CONSISTENT_MEMBER_ID from &rx_data. where CODE_VALUE in &RX_filter.)
              as _RX_Members
              INNER JOIN
              (select CONSISTENT_MEMBER_ID from &code_data. where CODE_VALUE in
&code_filter.)
              as _Code_Members
              ON _RX_Members.CONSISTENT_MEMBER_ID = _Code_Members.CONSISTENT_MEMBER_ID
      )
      as F
      ON code.CONSISTENT_MEMBER_ID = F.CONSISTENT_MEMBER_ID
      ;

       create table &Step1_output. as
       select CONSISTENT_MEMBER_ID
       from
               (select CONSISTENT_MEMBER_ID,
                            count(CODE_VALUE) as count
               from FILTERED_SERVICE_CODE
               where
                     SERVICE_BEGIN_DT >= date%str(%')&start_date.%str(%') AND
                     service_end_dt < date%str(%')&end_date.%str(%') AND
                     CODE_VALUE in &code_filter.
               group by CONSISTENT_MEMBER_ID) as subset1
      where count >= &diag_freq.
      ;

run;
quit;
```

Program 1: Custom Step - Program

INTERACTING WITH CUSTOM STEPS

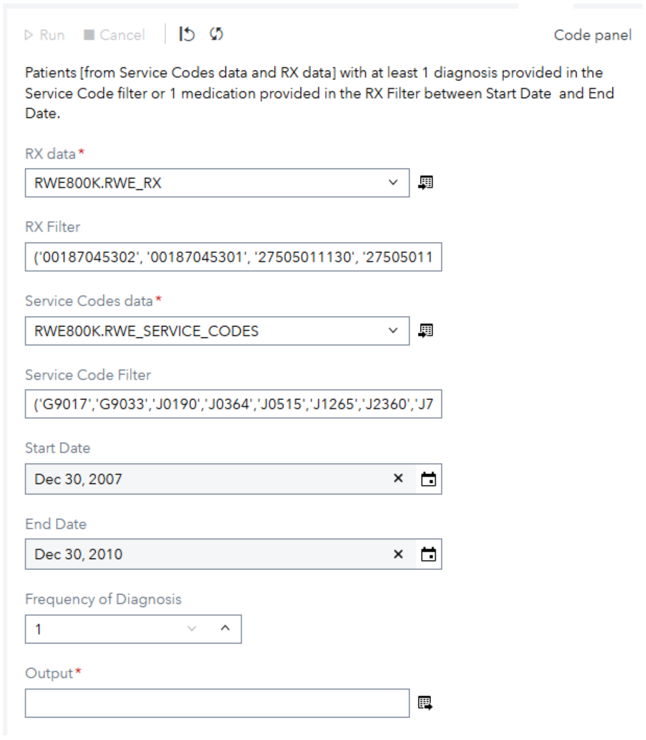
There are two primary ways to interact with Custom Steps once they have been built: as independent steps or as components within a SAS Flow. Running a Custom Step independently allows the specific process defined within it to be executed as a one-off action. However, this mode does not support scheduling. When a Custom Step is included within a Flow (even if the Flow consists of only a single step) it gains additional flexibility. The entire Flow can be executed either as a one-time run or scheduled as a recurring job.

INDEPENDENT

To interact with a Custom Step as a standalone component, right-click the step in question and select Open in a Tab. This action opens the Custom Step in a new tab, as shown in Display 2. In this view, you can see the output of the same program referenced in Display 1, with all associated controls rendered as interactive elements.

For instance, the RX Data control includes a link that lets users browse available libraries and datasets to select the desired input. Similarly, the calendar icon next to the Start Date field opens an interactive calendar for easy date selection.

It's important to note that in Display 2, the Run option is disabled because no output dataset has been selected. This is due to the output field being configured as a required variable within the Designer tab.



Patients [from Service Codes data and RX data] with at least 1 diagnosis provided in the Service Code filter or 1 medication provided in the RX Filter between Start Date and End Date.

RX data*
RWE800K.RWE_RX

RX Filter
('00187045302', '00187045301', '27505011130', '27505011

Service Codes data*
RWE800K.RWE_SERVICE_CODES

Service Code Filter
('G9017','G9033','J0190','J0364','J0515','J1265','J2360','J7

Start Date
Dec 30, 2007

End Date
Dec 30, 2010

Frequency of Diagnosis
1

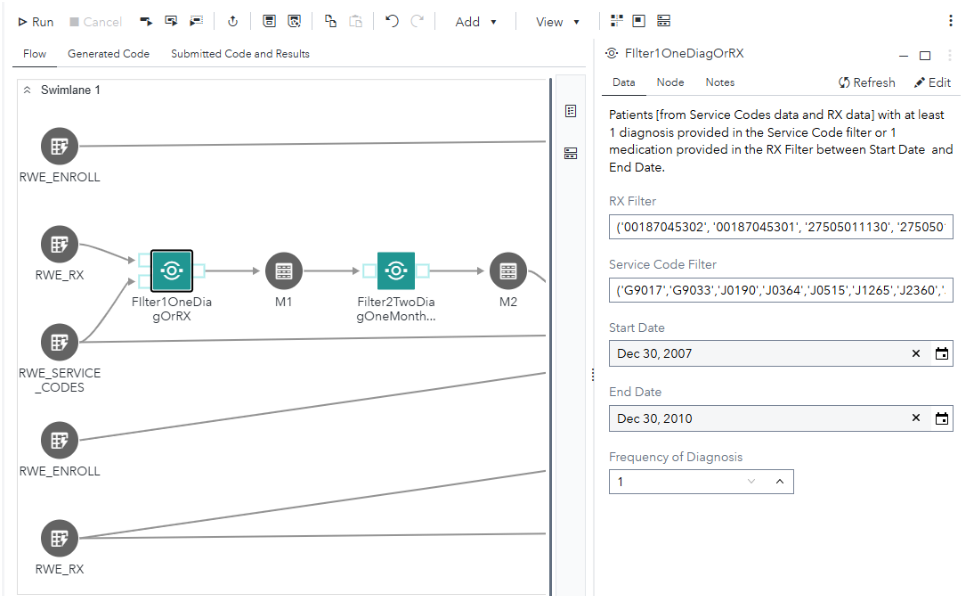
Output*

Display 2: Custom Step - Single Step

PROCESS FLOW

To use a Custom Step within a Flow, the user can either drag and drop the step into the Flow or right-click the step and select Add to Flow. In both cases, the Flow must already be open. The Custom Step can be selected from the user's program list, where it was originally saved, or, if it's stored in a shared location, accessed through the Step domain in SAS Studio under the Shared tab. This second option provides easy access to a centralized catalog of available Custom Steps.

The Custom Step has all the same interactive controls in a Flow as it does when used as a standalone component, as shown in Display 3.



Patients [from Service Codes data and RX data] with at least 1 diagnosis provided in the Service Code filter or 1 medication provided in the RX Filter between Start Date and End Date.

RX Filter

('00187045302', '00187045301', '27505011130', '275050

Service Code Filter

('G9017','G9033','J0190','J0364','J0515','J1265','J2360','

Start Date

Dec 30, 2007

End Date

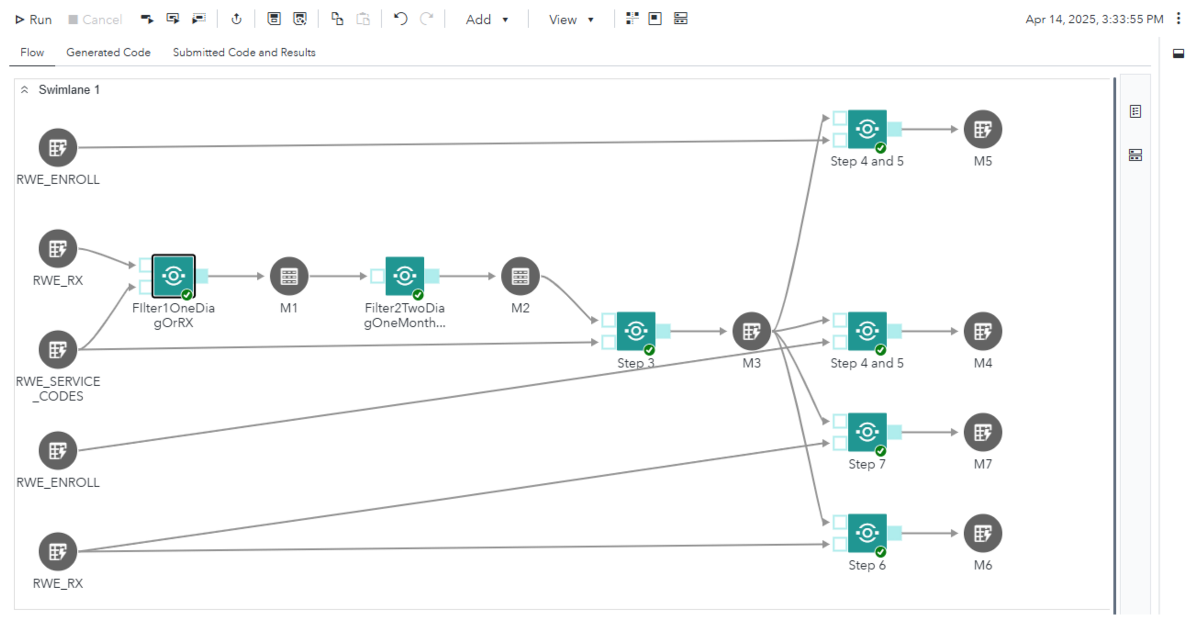
Dec 30, 2010

Frequency of Diagnosis

1

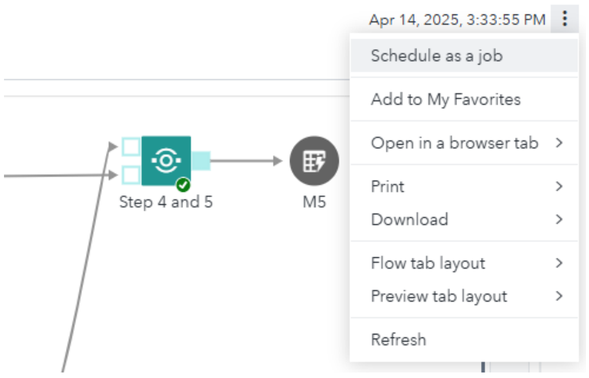
Display 3: Flow – Custom Step Interface

Display 4 shows an example of a completed Flow containing eight Custom Steps. Each step represents a different cohort filtering process. Notably, one of the steps is labeled Step 4 and 5, demonstrating how a single Custom Step can be reused in multiple contexts based on its configuration. This highlights the repeatability and flexibility of Custom Steps within a Flow.



Display 4: Flow

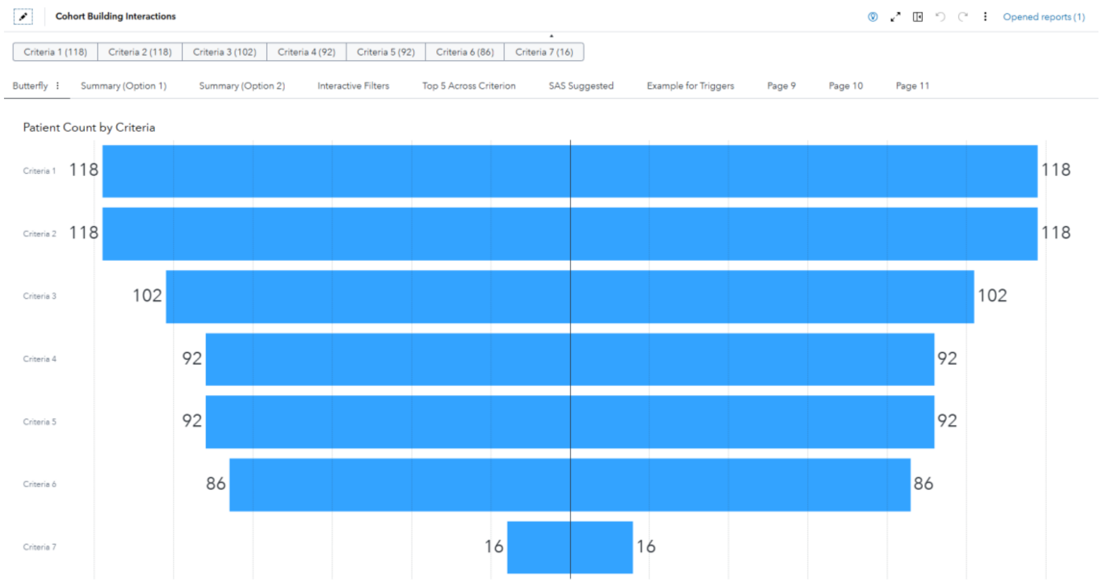
Finally, while the Flow and all of its internal Custom Steps can be run on an ad hoc basis by clicking the Run button in the top-right corner (as shown in Display 4), users also have the option to schedule the Flow as a background job. Display 5 illustrates this process: by selecting the ellipsis menu in the upperright corner and choosing Schedule as a Job, users can open the New Trigger window. This interface guides them through the options for configuring when and how the Flow should be executed automatically.



Display 5: Flow - Schedule as a Job

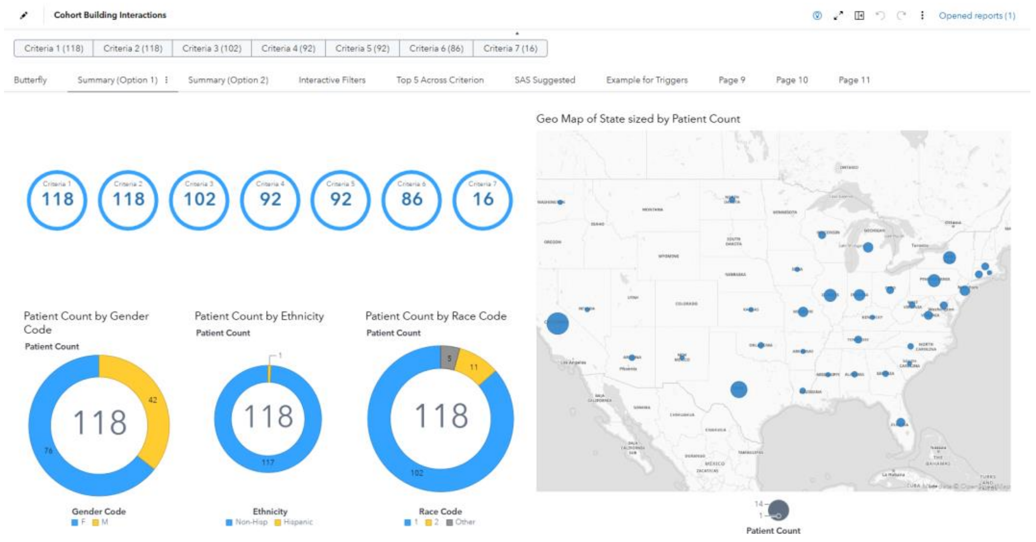
INTERACTING WITH DATA IN VISUAL ANALYTICS

Now that the different cohorts have been identified and partitioned, the resulting data has been loaded into a dashboard within Visual Analytics to enable easier interaction and exploration. Display 6 shows the first page of this dashboard, which features a butterfly chart illustrating patient attrition across each applied filter or criterion.



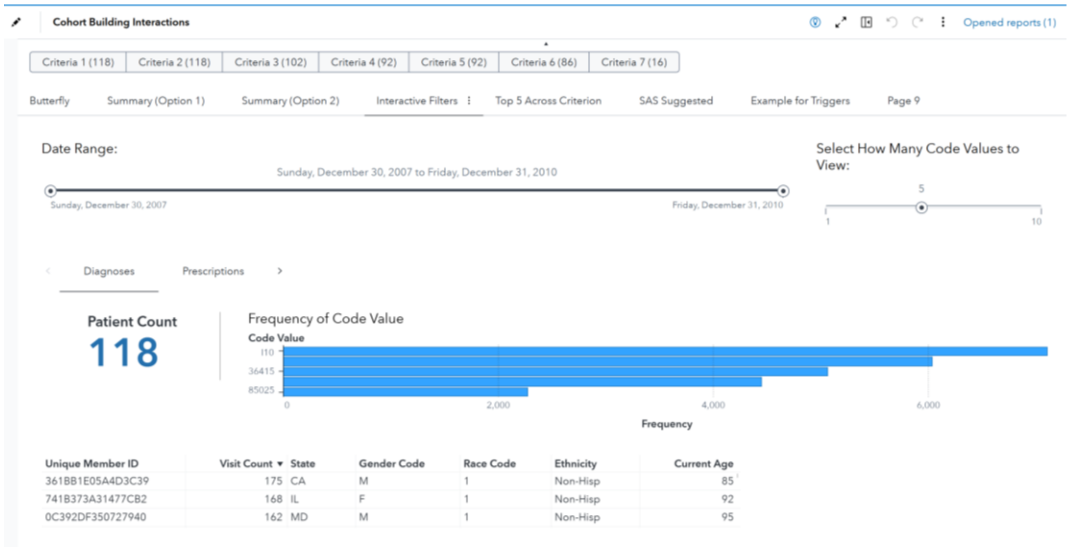
Display 6: Visual Analytics - Butterfly

Next, Display 7 begins to reveal how patient demographics can be affected by each criterion. The row of seven circular infographics, each representing counts for a specific criterion, serves as an interactive filter. When a user selects one, the demographic donut charts and the accompanying map update to reflect data specific to that criterion. Alternatively, users can filter by a specific criterion using the filter buttons located at the top of the dashboard.



Display 7: Visual Analytics - Summary

The Interactive Filters tab, shown in Display 8, demonstrates how users can dynamically slice and dice the data. The Date Range slider allows end users to explore how adjusting the start and end dates of the study period impacts the patient count and the top five diagnoses. While this adjustment does not re-run the underlying Custom Step filters, it offers a useful approximation for “what-if” scenarios. Additionally, another slider lets users modify the number of diagnosis codes displayed in the Frequency of Code Values bar chart, updating the chart to reflect the specified top selected count.



Unique Member ID	Visit Count ▾	State	Gender Code	Race Code	Ethnicity	Current Age
361BB1E05A4D3C39	175	CA	M	1	Non-Hisp	85
741B373A31477CB2	168	IL	F	1	Non-Hisp	92
0C392DF350727940	162	MD	M	1	Non-Hisp	95

Display 8: Visual Analytics - Interactive Filters

These are just a few examples on how Visual Analytics can help make insightful investigation of the data easy and accessible.

CONCLUSION

There are many options that researchers have at their disposal to efficiently and accurately define cohorts and accelerate evidence generation in support of drug development and integrated evidence packages. We demonstrated the ease of standardizing and reproducing cohort definitions to streamline the research process and make studies more efficient and scalable and facilitates data sharing and collaboration across institutions, which is crucial for large-scale observational studies.

REFERENCES

Mary Dolegowski's Git Hub. 2025. "RE-374". Accessed April 11, 2025. https://github.com/mary-dolegowski/RE-374

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors at:

Sherrine Eid, MPH
SAS Institute, Inc
Sherrine.Eid@sas.com
www.sas.com

Mary Dolegowski
SAS Institute, Inc
Mary.Dolegowski@sas.com
www.sas.com

Any brand and product names are trademarks of their respective companies.