PharmaSUG 2017 - Paper AD23

Supporting the CDISC Validation Life-Cycle with Microsoft Excel VBA

Eric Crockett, Chiltern International

ABSTRACT

Clinical research is increasingly based on standardized Clinical Data Interchange Standards Consortium (CDISC) data. Evaluating whether the data conforms to the applicable CDISC standards is required and is often an iterative process done over the course of a study. Documenting the explanations for issues that cannot be resolved and tracking the trends in conformance findings over the life-cycle of a study can easily turn into a burdensome manual process repeated with each Pinnacle 21 report. Utilizing Microsoft Excel Visual Basic for Applications (VBA) macro code that will be provided and discussed, successive validation reports can be compared and explanatory comments can be migrated. Leveraging this automated application can dramatically reduce the time spent tracking, evaluating and documenting conformance findings.

INTRODUCTION

Evaluating whether mapped data conforms to the applicable CDISC (SEND, SDTM or ADaM) standard is best done multiple times over the course of a study. Early reports confirm that initial mapping is sound but are often littered with findings triggered by dirty or incomplete data. Runs near the time of database lock are time-sensitive and any finding that cannot be addressed must be documented and included in the conformance section of the Study Data Reviewer's Guide (SDRG).

Efficiently managing the output from iterative Pinnacle 21 validation runs requires planning and can be effectively supported by an application. Over the course of a study, the general expectation is that issues based on dirty or incomplete data will resolve and that issues based in collection standards that are inconsistent with the CDISC requirements will remain. Starting with a general convention that the Issue Summary table of a Pinnacle 21 output will be used to: flag issues for corrective action, track increases or decreases in issue numbers over time, and document issues that are not expected to resolve as needed for inclusion in the SDRG. This paper will discuss a VBA macro that compares the Issue Summary table information betweem two Pinnacle 21 reports, color codes changes in the number of issues Found and migrates annotations.

OVERVIEW

A WELL-STRUCTURED TABLE

As noted above, the goal is to compare the Issue Summary tables from two Pinnacle 21 reports, generate (and color code) changes in number of issues Found and migrate notes and comments. To accomplish this, a row by row comparison between the reports is required.

Those who work with Microsoft Excel likely use the Control-key functions within Excel to scoot around a page quickly, select an entire range of cells or select individual cells. Excel VBA macros can be based on these same principles. The structure of a Pinnacle 21 report is entirely predictable and this makes it possible to implement a VBA macro that can work with and traverse the scaffolding of the Issue Summary table.

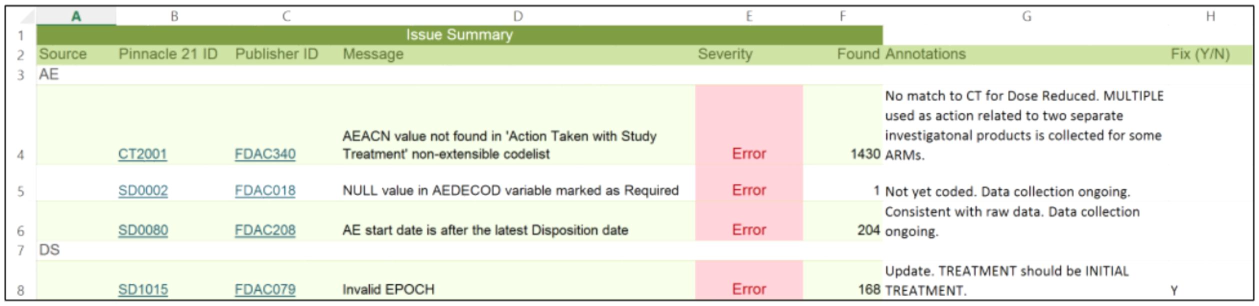
PROCESS SUMMARY

The macro uses the Date Modified dates to determine which report is the source of the comments (earlier date) and which is the target report (later date). The next task is to bring both Issue Summary tables into a new workbook so that nothing is modified directly. The structure of the original summary tables is updated to contain a dataset reference on each row so that it is possible to associate each issue with a dataset. Once this is accomplished on both Issue Summary tables, the program moves through each row on the annotated report table comparing them to each row in the new report and migrating the comments where exact matches are found for the Source and Message values. The number Found on the original annotated report is added to a cell comment and the color of the cell is changed to green (fewer issues) or red (more issues). No color change indicates the number of issues flagged was identical if the issue was present on both reports. Once the program finishes the last row from the original report, the original table is eliminated from the workbook and the final output is available and ready for review. If satisfactory, the comment columns can be copied and pasted as a group into the latest Pinnacle 21 report from both workbooks.

PROCESS

NEW WORKBOOK AND RELEVANT TABLES

The first task the VBA macro must do is determine which workbook has the earliest Date Modified date. The Issue Summary table from this workbook is copied into a new workbook first (see Figure 1). Then the macro copies in the Issue Summary table from the latest Date Modified report into the new workbook a number of cells to the right. Outputting to a new workbook allows the user to review and approve the final output and preserve the original Pinnacle 21 content.



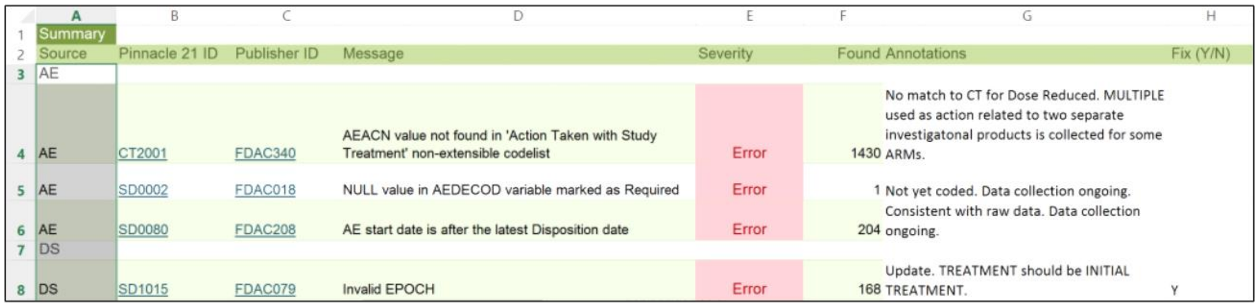
	A	B	C	D	E	F	G	H
1	Issue Summary							
2	Source	Pinnacle 21 ID	Publisher ID	Message	Severity	Found	Annotations	Fix (Y/N)
3	AE							
4		CT2001	FDAC340	AEACN value not found in 'Action Taken with Study Treatment' non-extensible codelist	Error	1430	No match to CT for Dose Reduced. MULTIPLE used as action related to two separate investigatonal products is collected for some ARMs.	
5		SD0002	FDAC018	NULL value in AEDECOD variable marked as Required	Error	1	Not yet coded. Data collection ongoing.	
6		SD0080	FDAC208	AE start date is after the latest Disposition date	Error	204	Consistent with raw data. Data collection ongoing.	
7	DS							
8		SD1015	FDAC079	Invalid EPOCH	Error	168	Update. TREATMENT should be INITIAL TREATMENT.	Y

Figure 1. Paste of Table from Original Pinnacle 21 Report

CARRY FORWARD DATASET HEADER INFORMATION

Nothing can proceed without having a simple way to associate each comment with a particular dataset. Each dataset header is actually a merged cell. Luckily, there is a VBA function that will unmerge all cells for a table. Doing this eliminates ambiguity in cell selection and navigation around the tables.

Each dataset name is carried forward (directionally downward) from the header with the dataset name (see the selected cells in Figure 2). The macro travels down column A filling in the dataset names. The last iteration is planned for the last row of the table. It is important to note here that the macro will not behave as expected if table filters are applied.

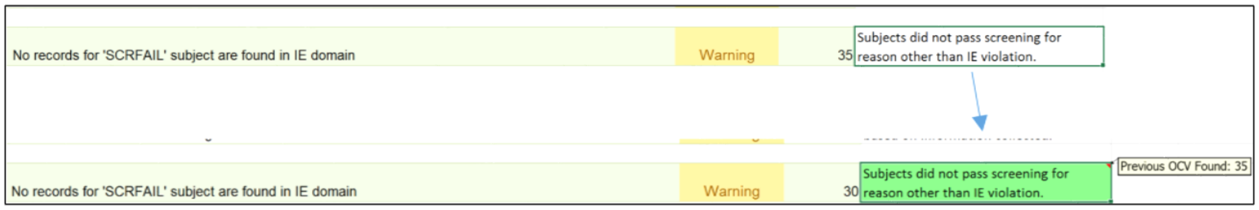


	A	B	C	D	E	F	G	H
1	Summary							
2	Source	Pinnacle 21 ID	Publisher ID	Message	Severity	Found	Annotations	Fix (Y/N)
3	AE							
4	AE	CT2001	FDAC340	AEACN value not found in 'Action Taken with Study Treatment' non-extensible codelist	Error	1430	No match to CT for Dose Reduced. MULTIPLE used as action related to two separate investigatonal products is collected for some ARMs.	
5	AE	SD0002	FDAC018	NULL value in AEDECOD variable marked as Required	Error	1	Not yet coded. Data collection ongoing.	
6	AE	SD0080	FDAC208	AE start date is after the latest Disposition date	Error	204	Consistent with raw data. Data collection ongoing.	
7	DS							
8	DS	SD1015	FDAC079	Invalid EPOCH	Error	168	Update. TREATMENT should be INITIAL TREATMENT.	Y

Figure 2. First Rows Depicting "Carry-Forward" Header Information

CYCLE THROUGH EACH PREVIOUS ISSUE & COPY AND PASTE EACH COMMENT

Now that each issue is associated with a dataset it is easy to cycle through each row of the original issues and comments. The original issues and comments are compared with the corresponding new issues iteratively row by row based on Source and Message. Then the number of issues is compared and the comments migrated. Comments are populated to the right of each issue and the number of issues previously Found is added as a cell comment of the cell with the annotation comment (see Figure 3).



No records for 'SCRFAIL' subject are found in IE domain	Warning	35	Subjects did not pass screening for reason other than IE violation.	
No records for 'SCRFAIL' subject are found in IE domain	Warning	30	Subjects did not pass screening for reason other than IE violation.	Previous OCV Found: 35

Figure 3. Migrating a Comment

FINAL OUTPUT

The original Issue Summary table is no longer needed once the last row of original issues is compared to the new report. The table is removed so that all that is left is the latest set of issues in a new workbook leaving both of the original documents untouched. It is easy to then accept or reject the output, especially if the output is not as expected, and copy and paste the comments as a group into the new Pinnacle 21 Report.

CONCLUSION

NUANCES AND CAUTIONS

While VBA macros are incredibly powerful and can be utilized to shed ourselves of some of the more mundane tasks, it is important to remember that VBA macros are final. You cannot undo the work of a VBA macro, in fact, you cannot undo anything that was changed prior to running the macro. For this reason, it is important to remember to save your work before using VBA macros.

There is another issue at hand here—malicious code. VBA is very powerful and can access the file system of your environment and system information such as your username. For this reason, it is a good general rule of thumb to never use VBA macros that are not company-approved or sufficiently vetted.

CLOSING

Excel VBA tools are an incredibly effective way to accomplish any iterative tasks that are well-defined. All that is required is a structured representation of information. VBA can be used to modify text within a cell, do statistics, or create copy and paste ready tables for a Study Data Reviewer’s Guide. VBA solutions have proven to be an effective tool in the life-cycle maintenance of clinical data.

ACKNOWLEDGMENTS

A special thanks to Steven Kirby, who helped make this paper possible through his support of my Excel VBA endeavors.

RECOMMENDED READING

- *WiseOwlTutorials – YouTube.com*

APPENDIX

CODE

```
Sub UpdateNewOpenCDISC()

    Application.ScreenUpdating = False
    Dim FirstWorkBook, MidBook, LastWorkBook, WorkingBook As Workbook

    Dim SingleCell, ListOfCells, CompareCell, ListOfCompareCells As Range
    Dim CompareDataset, CompareCode As String
    Dim CompareCount As Integer

    Dim Ldate As Date
    Dim Fdate As Date
    Dim Mdate As Date

    Dim ActivePath As String
    Dim NumBooks As Integer
    Dim i As Integer

    Dim xComm As Comment

    Set FirstWorkBook = ActiveWorkbook
    Set MidBook = ActiveWorkbook
    Set LastWorkBook = ActiveWorkbook

    ActivePath = ActiveWorkbook.Path
    Fdate = DateValue("Jan 1, 2100 00:00:00 AM")
    Ldate = DateValue("Jan 1, 1950 00:00:00 AM")

    NumBooks = Workbooks.Count

    For i = 1 To NumBooks

        Workbooks(i).Activate
        Set MidBook = Workbooks(i)
        Mdate = FileDateTime(Workbooks(i).FullName)

'        Workbooks("Start-Up").Activate
        MidBook.Activate

        If Fdate > Mdate And ActiveWorkbook.Name <> "Start-Up.xlsm" Then
            Fdate = Mdate
            Set FirstWorkBook = MidBook
        End If
        If Ldate < Mdate And ActiveWorkbook.Name <> "Start-Up.xlsm" Then
            Ldate = Mdate
            Set LastWorkBook = MidBook
        End If

    Next i
    FirstWorkBook.Worksheets("Issue Summary").Activate

    Range("A1").End(xlDown).End(xlDown).End(xlDown).Offset(1, 0).Select
    Selection.CurrentRegion.Select
    Selection.Copy

    Set WorkingBook = Workbooks.Add
    Range("A1").Activate
    ActiveCell.PasteSpecial (xlPasteColumnWidths)
    ActiveCell.PasteSpecial
    Range("A1").CurrentRegion.MergeCells = False
    Set SingleCell = Range("A1").End(xlDown).Offset(1, 0)

    Do
        If SingleCell.Value = "" Then
            SingleCell.Value = SingleCell.Offset(-1, 0).Value
        End If
        Set SingleCell = SingleCell.Offset(1, 0)
    Loop Until SingleCell.Offset(0, 1).Value = "" And SingleCell.Offset(1, 1).Value
= "" And SingleCell.Offset(1, 0).Value = ""

    LastWorkBook.Worksheets("Issue Summary").Activate

    Range("A1").End(xlDown).End(xlDown).End(xlDown).Offset(1, 0).Select
    Selection.CurrentRegion.Select
    Selection.Copy

    WorkingBook.Activate
    Range("U1").Activate
    ActiveCell.PasteSpecial (xlPasteColumnWidths)
    ActiveCell.PasteSpecial
    Range("U1").CurrentRegion.MergeCells = False

    Set SingleCell = Range("U1").End(xlDown).Offset(1, 0)

    Do
        If SingleCell.Value = "" Then
            SingleCell.Value = SingleCell.Offset(-1, 0).Value
        End If
        Set SingleCell = SingleCell.Offset(1, 0)
    Loop Until SingleCell.Offset(0, 1).Value = "" And SingleCell.Offset(1, 1).Value
= "" And SingleCell.Offset(1, 0).Value = ""
    Range("A1").EntireRow.Delete

    Set ListOfCells = Range("A3", Range("A3").End(xlDown))

    For Each SingleCell In ListOfCells
        Set ListOfCompareCells = FindRange(SingleCell.Value)
        For Each CompareCell In ListOfCompareCells

            If SingleCell.Offset(0, 3).Value = CompareCell.Offset(0, 3).Value And
CompareCell.Offset(0, 1) <> "" Then

                CompareCell.Offset(0, 6).Value = SingleCell.Offset(0, 6).Value
                CompareCell.Offset(0, 7).Value = SingleCell.Offset(0, 7).Value
                CompareCell.Offset(0, 8).Value = SingleCell.Offset(0, 8).Value
                CompareCell.Offset(0, 9).Value = SingleCell.Offset(0, 9).Value
                CompareCell.Offset(0, 10).Value = SingleCell.Offset(0, 10).Value
                CompareCell.Offset(0, 11).Value = SingleCell.Offset(0, 11).Value
                CompareCell.Offset(0, 12).Value = SingleCell.Offset(0, 12).Value
                CompareCell.Offset(0, 13).Value = SingleCell.Offset(0, 13).Value
                CompareCell.Offset(0, 14).Value = SingleCell.Offset(0, 14).Value
                i = WorksheetFunction.Min(Abs(SingleCell.Offset(0, 5).Value -
CompareCell.Offset(0, 5).Value), 255)
                If SingleCell.Offset(0, 5).Value > CompareCell.Offset(0, 5).Value
And (CompareCell.Offset(0, 6) <> "" Or CompareCell.Offset(0, 7) <> "") Then
                    CompareCell.Offset(0, 6).Interior.Color = RGB(133, 255, 133)
                    CompareCell.ClearComments
                    CompareCell.Offset(0, 6).AddComment "Previous OCV Found: " &
SingleCell.Offset(0, 5).Value
                End If
                If SingleCell.Offset(0, 5).Value < CompareCell.Offset(0, 5).Value
And (CompareCell.Offset(0, 6) <> "" Or CompareCell.Offset(0, 7) <> "") Then
                    CompareCell.Offset(0, 6).Interior.Color = RGB(255, 133, 133)
                    CompareCell.ClearComments
                    CompareCell.Offset(0, 6).AddComment "Previous OCV Found: " &
SingleCell.Offset(0, 5).Value

                End If
            End If
        Next CompareCell
    Next SingleCell
    For Each xComm In Application.ActiveSheet.Comments
        xComm.Shape.TextFrame.AutoSize = True
    Next

    Range("A1", "T1").EntireColumn.Delete
    Application.ScreenUpdating = True

End Sub


Function FindRange(Dataset As String) As Range

    Dim SingleCell, ListOfCells, FirstCell, LastCell As Range
    Set ListOfCells = Range("U3", Range("U3").End(xlDown))
    Set FirstCell = Range("T1")
    Set LastCell = Range("T2")

    For Each SingleCell In ListOfCells
        If SingleCell.Value = Dataset And FirstCell.Value = "" Then
            Set FirstCell = SingleCell
        ElseIf SingleCell.Value = Dataset And FirstCell.Value <> "" Then
            Set LastCell = SingleCell
        End If
    Next SingleCell
    Set FindRange = Range(FirstCell, LastCell)

End Function
```

CONTACT INFORMATION

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

Eric Crockett
Chiltern International
eric.crockett@chiltern.com