ARD Generator

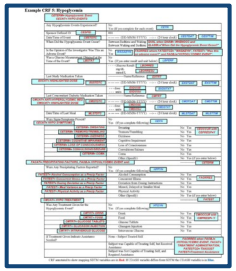
Metadata-driven analysis using the Analysis Results Standard

COSA Spotlight 12 December 2023

Karl Wallendszus, Oxford Population Health

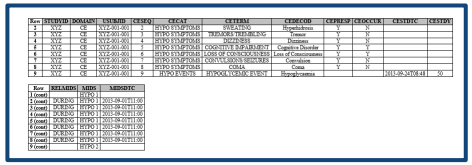
cdisc

CDISC Foundational Standards



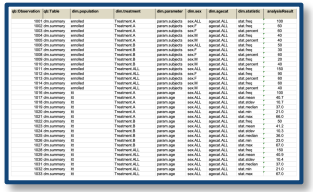
Data Collection

CDASH



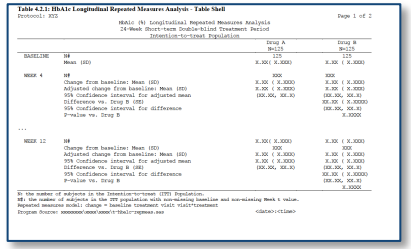
Data Aggregation

SDTM



Analysis

ADaM



Results

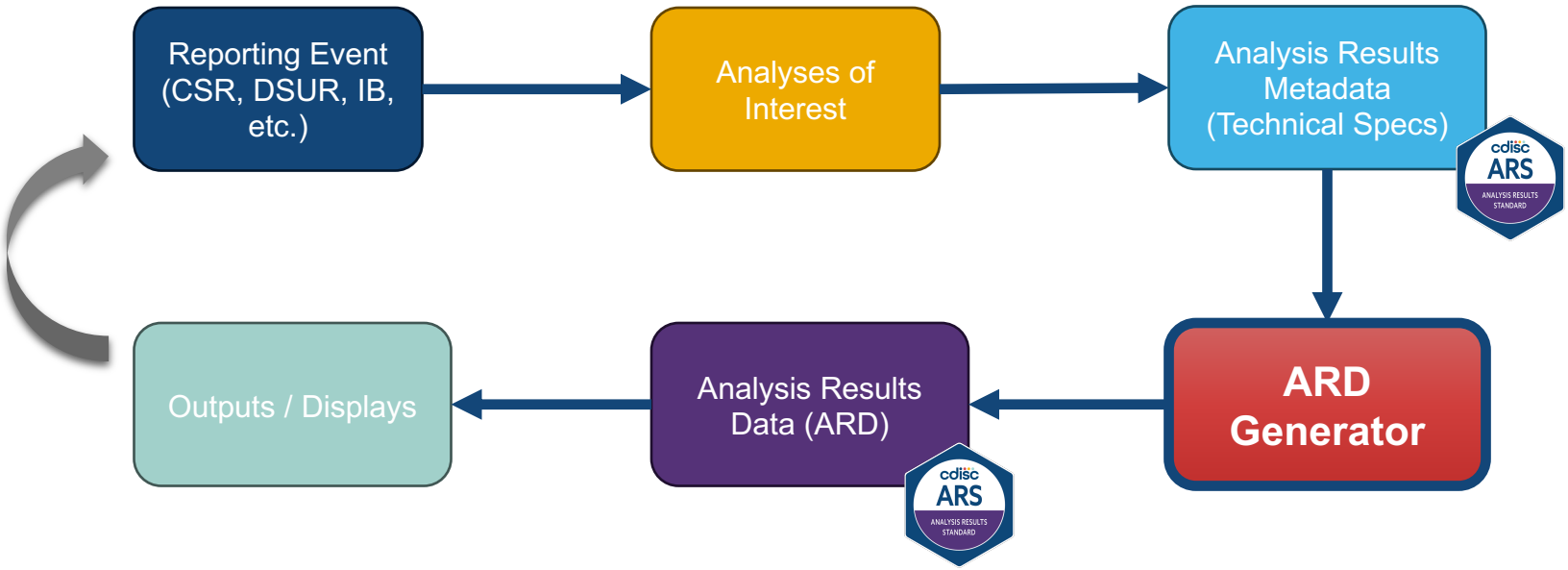
???

Table 4.2.2: HbA1c Longitudinal Repeated Measures Analysis Results Metadata

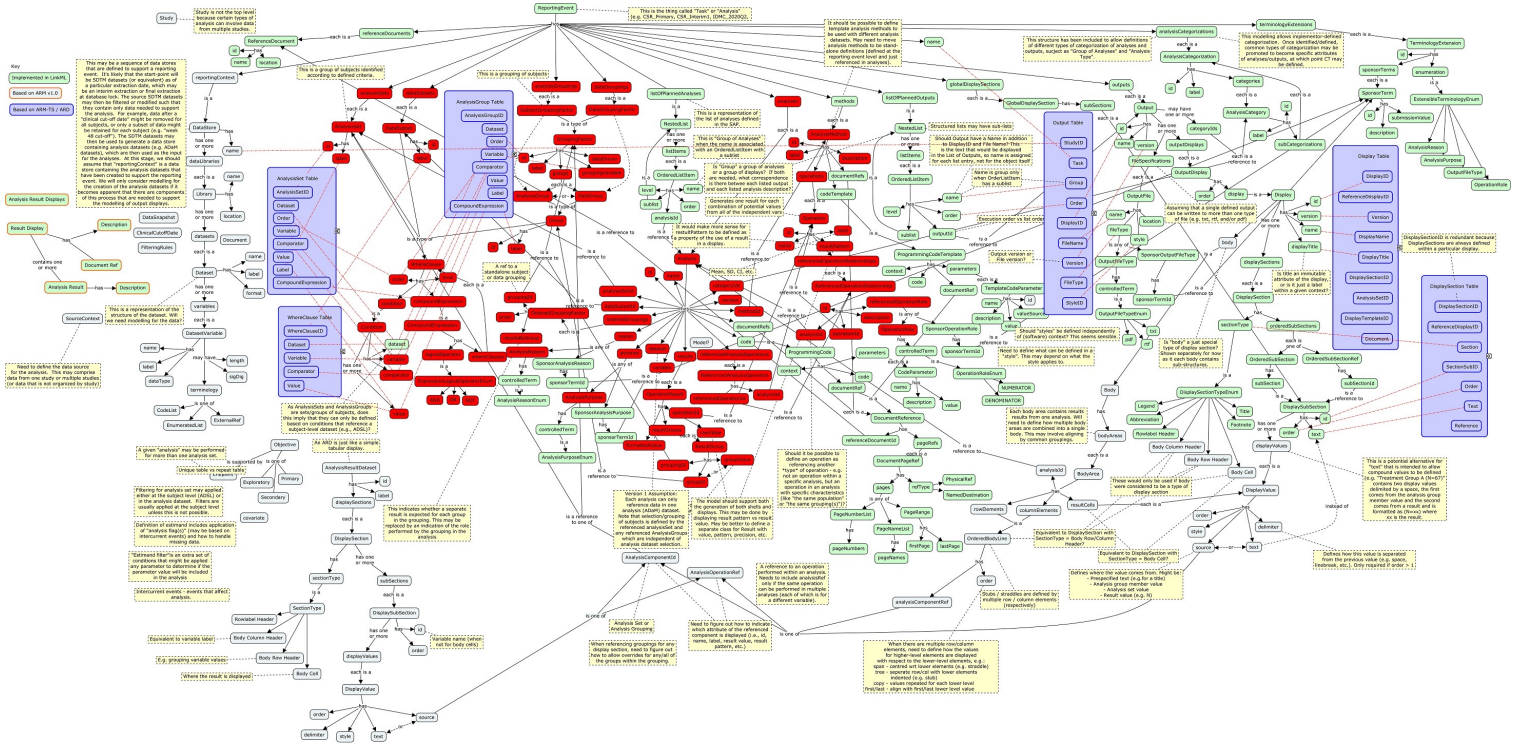
Metadata Field	Metadata
DISPLAY IDENTIFIER	Table 4.2.1/Figure 4.2.1
DISPLAY NAME	Mean Change from Baseline in HbA1c (Percent) Longitudinal Repeated Measures Analysis, 24-Week Short-term Double-blind Treatment Period, Intention-to-treat Population
RESULT IDENTIFIER	Treatment difference results (LSMean, confidence interval, p-value)
PARAM	HbA1c (%)
PARAMCD	HBA1C
ANALYSIS VARIABLE	CHG (Change from baseline)
ANALYSIS REASON	SPECIFIED IN SAP
ANALYSIS PURPOSE	PRIMARY OUTCOME MEASURE
ANALYSIS DATASET	ADHBA1C

ARM for Define.XML

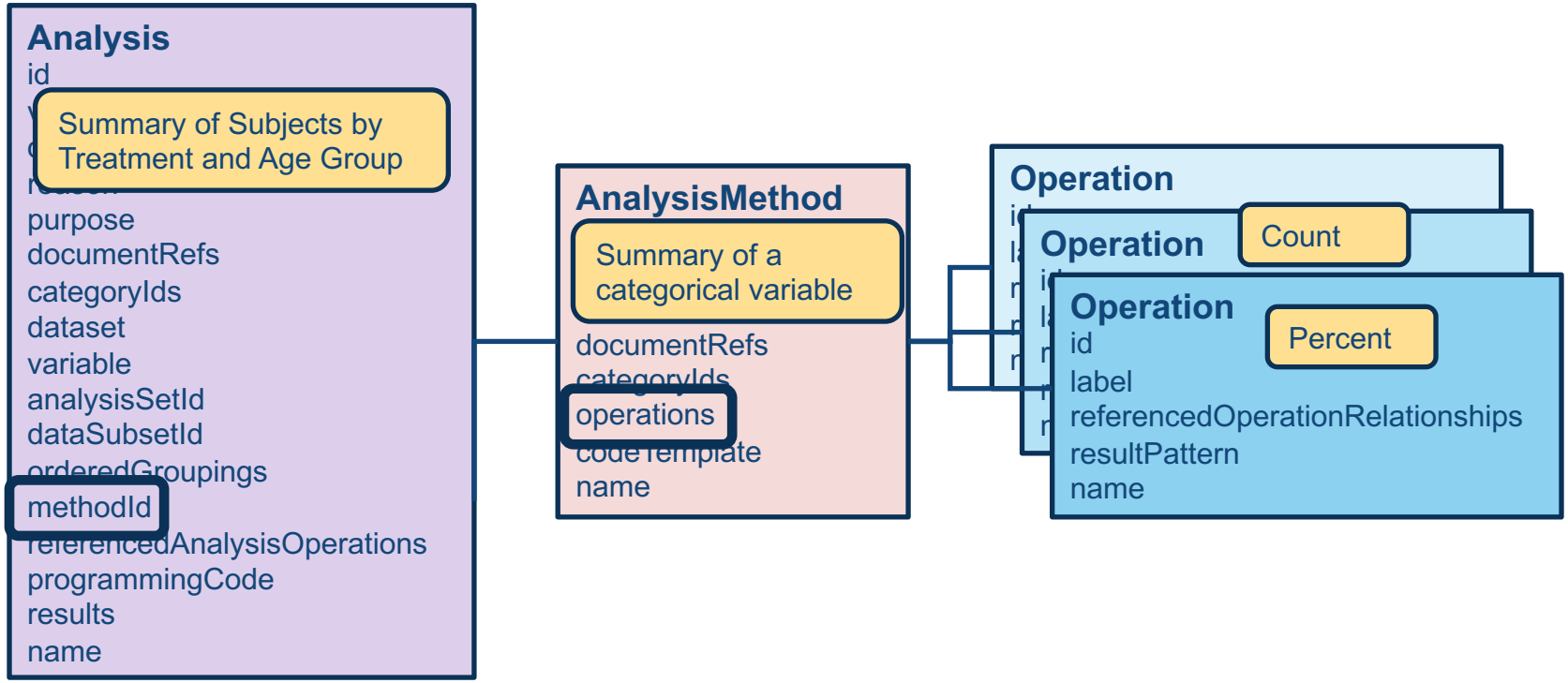
ARS Model Supported Workflow and Entry Points



ARS model



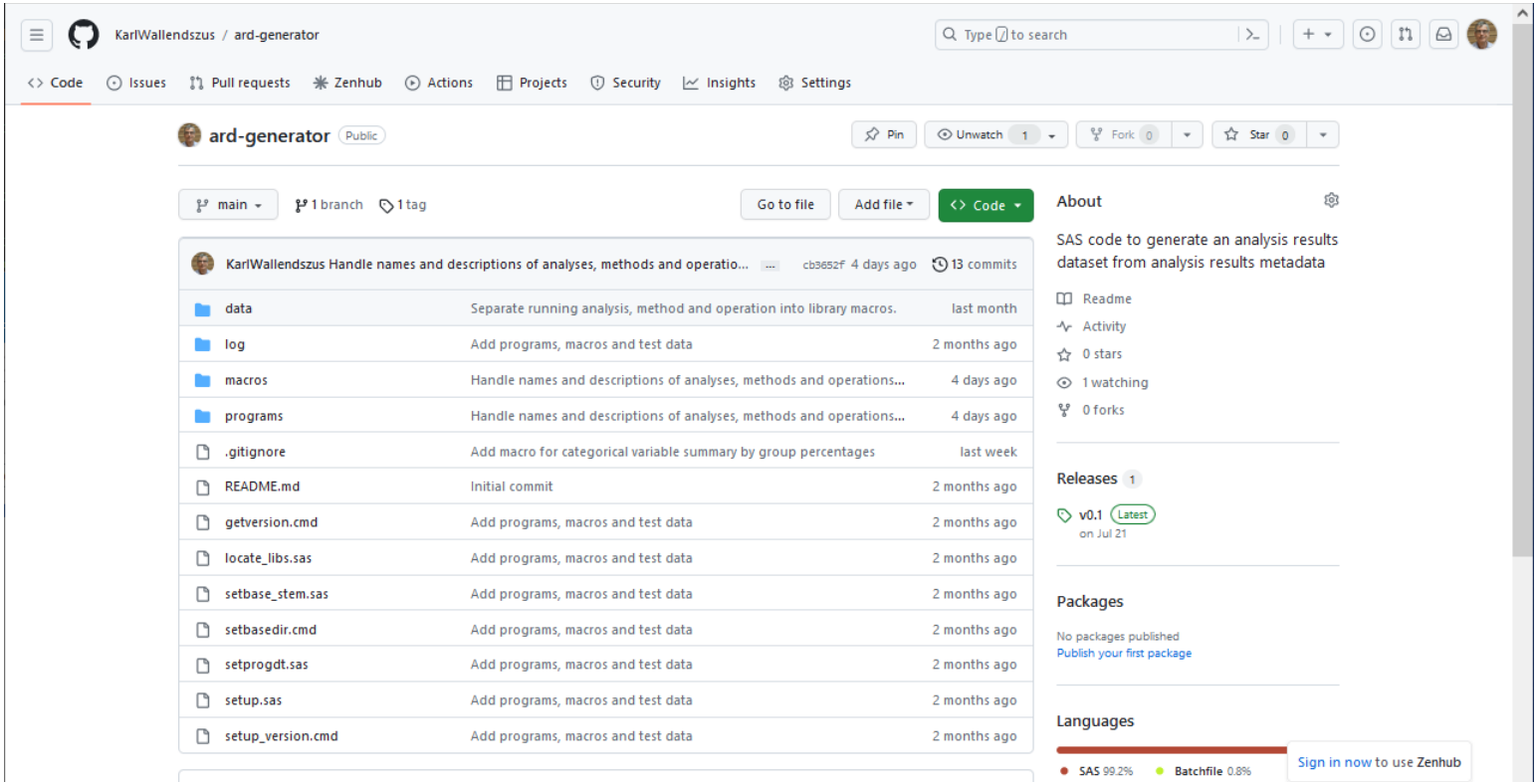
Analysis – Method - Operation



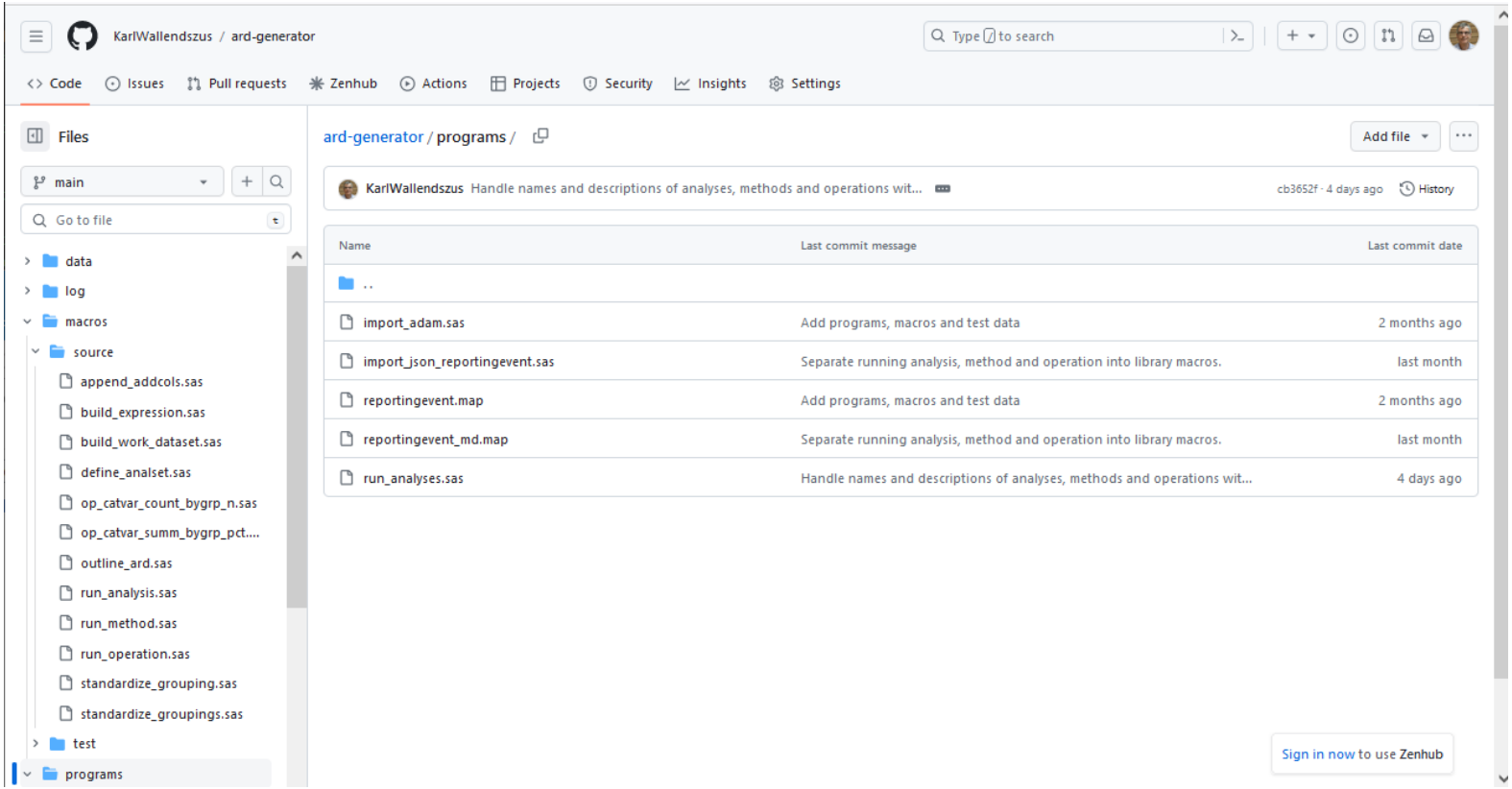
Structure of code

- Written in SAS
- Ingest ARS metadata from JSON → SAS datasets
- Process analyses specified in metadata
- Macro library
  - Analysis
  - Method
  - Operation
  - Particular operations
- Output to Analysis Results Dataset (ARD)

Structure of code



Structure of code



Run analyses

```
*******************************************************************************;
* Main code
*******************************************************************************;

* Create dataset for expressions in SAS syntax for conditions;
data jsonmd.expressions;
    length id $40 label $200 dsconds $32 expression $200;

    * Derive expressions in SAS syntax for conditions;

    * Run analyses;
    %run_planned_analyses(mdlib=jsonmd, datalib=adam, ardlib=ard);
    /*
    %run_analysis(mdlib=jsonmd, datalib=adam, ardlib=ard,
        analid=An01_05_SAF_Summ_ByTrt, debugfl=Y);
    %run_analysis(mdlib=jsonmd, datalib=adam, ardlib=ard,
        analid=An03_02_AgeGrp_Summ_ByTrt, debugfl=Y);
    %run_analysis(mdlib=jsonmd, datalib=adam, ardlib=ard,
        analid=An03_05_Race_Summ_ByTrt, debugfl=Y);
    %run_analysis(mdlib=jsonmd, datalib=adam, ardlib=ard,
        analid=An07_09_Soc_Summ_ByTrt, debugfl=Y);
    */
%run_
    a
%run_
    a
%run
    analid=An07_09_Soc_Summ_ByTrt, debugfl=Y);
*/
```

Run analyses

```
/**
* Run all planned analyses.
*
* @param mdlib      Library containing metadata datasets.
* @param datalib    Library containing data to be analysed.
* @param ardlib     Library containing analysis results datasets.
* @param debugfl    Debug flag (Y/N).
*/
%macro run_planned_analyses ( mdlib=, datalib=, ardlib=, debugfl=N );

    * Get list of planned analyses;
    %local analids;
    proc sql;
        select distinct analysisid into :analids separated by '|'
            from &mdlib..listofplannedanalyses
            where analysisid ^= ''
            order by 1;
    quit;

    * Loop through analyses;
    %local ianal analid;
    %let ianal = 1;
    %do %while(%scan(&analids., &ianal., '|') ne );
        %let analid = %scan(&analids., &ianal., '|');

        * Run this analysis;
        %run_analysis(mdlib=&mdlib., datalib=&datalib., ardlib=&ardlib.,
            analid=&analid., debugfl=&debugfl.);

        %let ianal = %eval(&ianal.+1);
    %end;

%mend run_planned_analyses;
```

Run a single analysis

```
%macro run_analysis ( mdlib=, datalib=, ardlib=, analid=, debugfl=N );

    * Get analysis details;
    %local analname analdesc analver analreas analpurp analds analvar analsetid
        datasubsetid methid
        ngroupings groupingords groupingids groupingresbys
        nrefops refopords refoprelids refopanalids
        docrefs catids;
    proc sql;
        select name, description, version, reason, purpose, dataset, variable,
                    analysisSetId, dataSubsetId, method_id
                into :analname, :analdesc, :analver, :analreas, :analpurp,
                    :analds, :analvar, :analsetid, :datasubsetid, :methid
            from &mdlib..analyses
            where id = "&analid.";
        select count(*) into :ngroupings
            from &mdlib..analysesordgroupings
            where id = "&analid.";
        select order, groupingId, resultsByGroup
            into :groupingords separated by '|', :groupingids separated by '|',
                :groupingresbys separated by '|'
            from &mdlib..analysesordgroupings
            where id = "&analid."
            order by 1;
        select count(*) into :nrefops
            from &mdlib..analysesrefoperations
            where id = "&analid.";
        select order, referencedOperationRelationshipI, analysisId
            into :refopords separated by '|', :refoprelids separated by '|',
                :refopanalids separated by '|'
            from &mdlib..analysesrefoperations
            where id = "&analid."
            order by 1;
    quit;
```

Run a single analysis

```
* Show analysis details;
%put NOTE: ================================================================================;
%put NOTE: Analysis &analid.: &analname.;
%put NOTE:   Description: %bquote(&analdesc.);
%put NOTE:   Version: &analver.;
%put NOTE:   Reason: &analreas.;
%put NOTE:   Purpose: &analpurp.;
%put NOTE:   Dataset: &analds.;
%put NOTE:   Variable: &analvar.;
%put NOTE:   Analysis set: &analsetid.;
%put NOTE:   Data subset: &datasubsetid.;
%put NOTE:   Method: &methid.;
%do ig = 1 %to &ngroupings.;
    %put NOTE:   Grouping &&groupingord&ig.: &&groupingid&ig. (result by group: &&groupingresby&ig.);
%end;
%do ir = 1 %to &nrefops.;
    %put NOTE:   Referenced operation &&refopord&ir.:;
    %put NOTE:     Relationship &&refoprelid&ir.;
    %put NOTE:     Analysis: &&refopanalid&ir.;
%end;

* Run the analysis method;
%run_method(mdlib=&mdlib., datalib=&datalib., ardlib=&ardlib.,
    methid=&methid., analid=&analid., analsetid=&analsetid.,
    datasubsetid=&datasubsetid., analds=&analds., analvar=&analvar.,
    groupingids=&groupingids., debugfl=&debugfl.);

* Write completion message to log;
%put NOTE: Analysis &analid. completed;
%put NOTE: ================================================================================;
```

Run an analysis method

```
%macro run_method ( mdlib=, datalib=, ardlib=, methid=, analid=,
    analsetid=, datasubsetid=, analds=, analvar=, groupingids=, debugfl=N );

    %* Get operation details;
    %local methid methname methlabel methdescr
        noperations opords opids;
    proc sql;
        select name, label, description into :methname, :methlabel, :methdescr
            from &mdlib..analysismethods
            where id = "&methid.";
        select count(*) into :noperations
            from &mdlib..methodoperations
            where id = "&methid.";
        select operation_order, operation_id
                into :opords separated by '|', :opids separated by '|'
            from &mdlib..methodoperations
            where id = "&methid."
            order by 1;
    quit;
```

Run an analysis method

```
%* Build a work dataset including all relevant variables;
%build_work_dataset(mdlib=&mdlib., datalib=&datalib., analds=&analds.,
    analvar=&analvar., analsetid=&analsetid.,
    datasubsetid=&datasubsetid., groupingids=&groupingids.,
    fmtlib=&ardlib., debugfl=&debugfl.);

%* Loop through analysis operations;
%do iop = 1 %to &noperations.;

    %* Execute this operation;
    %run_operation(mdlib=&mdlib., datalib=&datalib.,
        opid=&&opid&iop., methid=&methid., analid=&analid.,
        analsetid=&analsetid., datasubsetid=&datasubsetid.,
        groupingids=&groupingids., analds=workds, analvar=&analvar.,
        ard=&ardlib..ard, debugfl=&debugfl.);

%end;
```

Run an analysis operation

```
%* Create a work version of the ARD with a row for each expected result;
%outline_ard(ardlib=work, mdlib=&mdlib., analid=&analid., opid=&opid.,
    groupingids=&groupingids., dsin=&analds., dsout=work.ard);
```

	resultGroup1_groupLabel	resultGroup1_groupValue	resultGroup2_groupingId	resultGroup2_groupId	resultGroup2_groupLabel	resultGroup2_groupValue	rawValue
1	Placebo	AnlsGrouping_01_Trt_1	AnlsGrouping_03_AgeGp	AnlsGrouping_03_AgeGp_1	< 65 years	AnlsGrouping_03_AgeGp_1	
2	Xanomeline Low Dose	AnlsGrouping_01_Trt_2	AnlsGrouping_03_AgeGp	AnlsGrouping_03_AgeGp_1	< 65 years	AnlsGrouping_03_AgeGp_1	
3	Xanomeline High Dose	AnlsGrouping_01_Trt_3	AnlsGrouping_03_AgeGp	AnlsGrouping_03_AgeGp_1	< 65 years	AnlsGrouping_03_AgeGp_1	
4	Placebo	AnlsGrouping_01_Trt_1	AnlsGrouping_03_AgeGp	AnlsGrouping_03_AgeGp_2	≥ 65 years	AnlsGrouping_03_AgeGp_2	
5	Xanomeline Low Dose	AnlsGrouping_01_Trt_2	AnlsGrouping_03_AgeGp	AnlsGrouping_03_AgeGp_2	≥ 65 years	AnlsGrouping_03_AgeGp_2	
6	Xanomeline High Dose	AnlsGrouping_01_Trt_3	AnlsGrouping_03_AgeGp	AnlsGrouping_03_AgeGp_2	≥ 65 years	AnlsGrouping_03_AgeGp_2	

```
        groupingids=&groupingids., dsin=&analds., dsout=work.ard,
        debugfl=&debugfl.);
%end;
%else %if &opid. = Mth01_CatVar_Summ_ByGrp_2_pct %then %do;
```

Run an analysis operation

```
%op_catvar_summ_bygrp_pct(analid=&analid., methid=&methid., opid=&opid.,
    groupingids=&groupingids.,
    num_analid=&num_analid., num_opid=&&relopid&irel_num.,
    den_analid=&den_analid., den_opid=&&relopid&irel_den.,
```

	resultGroup1_groupLabel	resultGroup1_groupValue	resultGroup2_groupingId	resultGroup2_groupId	resultGroup2_groupLabel	resultGroup2_groupValue	rawValue
1	Placebo	AnlsGrouping_01_Trt_1	AnlsGrouping_03_AgeGp	AnlsGrouping_03_AgeGp_1	< 65 years	AnlsGrouping_03_AgeGp_1	14
2	Xanomeline Low Dose	AnlsGrouping_01_Trt_2	AnlsGrouping_03_AgeGp	AnlsGrouping_03_AgeGp_1	< 65 years	AnlsGrouping_03_AgeGp_1	8
3	Xanomeline High Dose	AnlsGrouping_01_Trt_3	AnlsGrouping_03_AgeGp	AnlsGrouping_03_AgeGp_1	< 65 years	AnlsGrouping_03_AgeGp_1	11
4	Placebo	AnlsGrouping_01_Trt_1	AnlsGrouping_03_AgeGp	AnlsGrouping_03_AgeGp_2	≥ 65 years	AnlsGrouping_03_AgeGp_2	72
5	Xanomeline Low Dose	AnlsGrouping_01_Trt_2	AnlsGrouping_03_AgeGp	AnlsGrouping_03_AgeGp_2	≥ 65 years	AnlsGrouping_03_AgeGp_2	76
6	Xanomeline High Dose	AnlsGrouping_01_Trt_3	AnlsGrouping_03_AgeGp	AnlsGrouping_03_AgeGp_2	≥ 65 years	AnlsGrouping_03_AgeGp_2	73

```
%end;

* Append work ARD to main ARD;
%append_addcols(dsbase=&ard., dsnew=work.ard);
```

Future development

- Handle more operations
- Implement display patterns
- Output formats
  - JSON
  - XPT?

Contact Details

Karl Wallendszus
Clinical Trial Service Unit, Oxford Population Health
University of Oxford, UK
karl.wallendszus@ndph.ox.ac.uk

https://github.com/KarlWallendszus/ard-generator