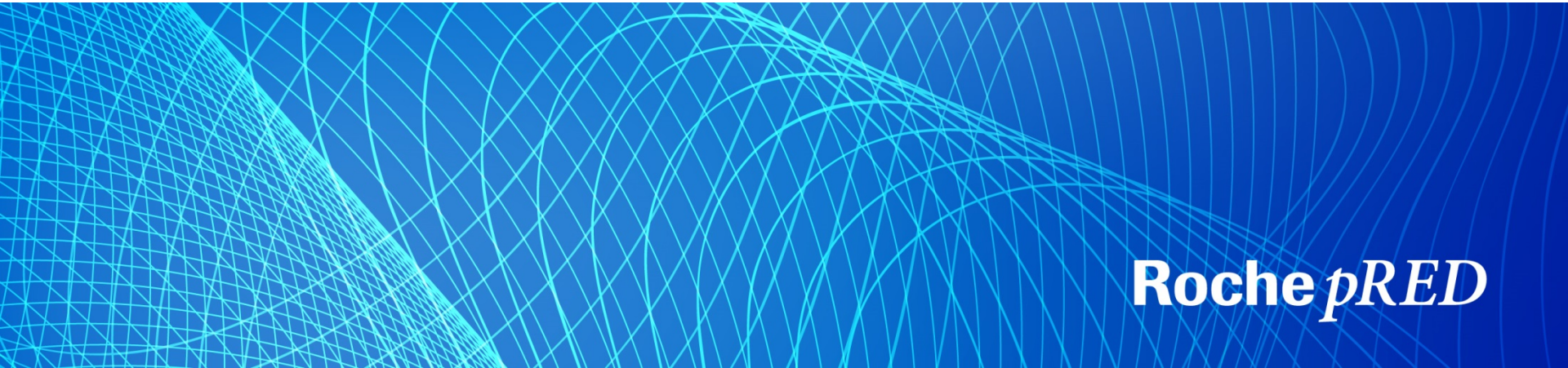
# Reducing Dataset Merges with Data Driven Formats

Paul Grimsey

Phuse 2017

CT01



# The Problem




= dataset

In dataset creation, if each step is dependent on the previous step, this can lead to complexity in the code and difficulty in it's maintenance.

# The Problem

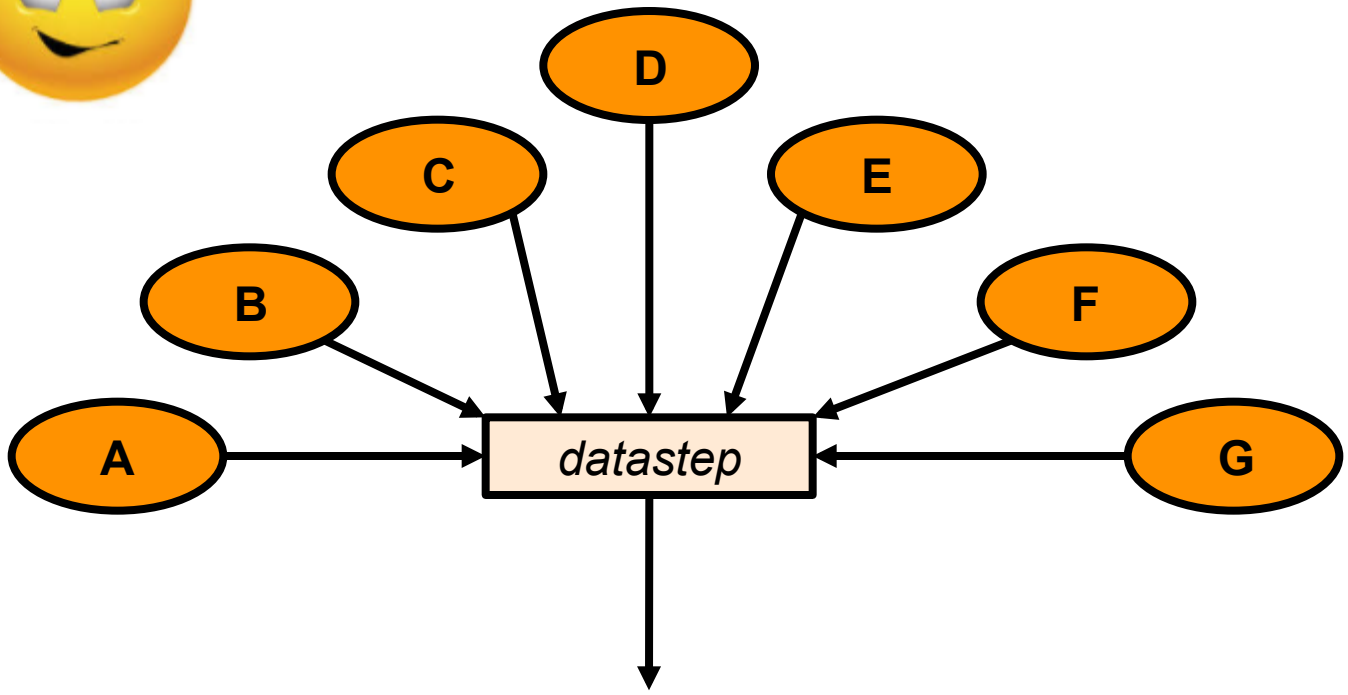



= dataset

An unexpected change to the data in one step could have consequences to the downstream steps.

# A Solution

If we can reduce the dependencies by separating out steps then our code can be easier to maintain and understand.




= dataset

With formats we can do this for 1 for 1 merges

# How to Merge Data Using Formats

**1)** Input dataset: WORK.AE

| USUBJID | AETERM |
|---|---|
| 12345-001-001 | Headache |
| 12345-001-001 | Dizziness |
| 12345-001-002 | Nausea |
| 12345-001-004 | Happiness |

**2)** Format containing information we want to add to WORK.AE:

```
proc format;
  value $cohort
  "12345-001-001" = "A"
  "12345-001-002" = "A"
  "12345-001-003" = "B"
  "12345-001-004" = "B"
  ;
run;
```

Applying the format adds **COHORT** to the **AE** dataset using **USUBJID** as the merge key

**3)** Example syntax for adding COHORT column to WORK.AE with a format.

```
data aecohort;
  set ae;

 cohort = put(usubjid, $cohort.);

run;
```

**4)** Output dataset: WORK.AECOHORT

| USUBJID | AETERM | COHORT |
|---|---|---|
| 12345-001-001 | Headache | A |
| 12345-001-001 | Dizziness | A |
| 12345-001-002 | Nausea | A |
| 12345-001-004 | Happiness | B |

# Other Benefits to Using Formats for Merging

- ☺ No sorting required at merge
- ☺ Faster processing times (than MERGE or SQL join
- ☺ Only join what is required
- ☺ Reusable
- ☺ Keys are unique
- ☺ No warnings for different length variables
- ☺ Conditional merging allowed
- ☺ ‘Other’ values can be specified

# CNTLIN Dataset

The **CNTLIN** option in **PROC FORMAT** allows creation of a format from a dataset

```
proc format library = work.formats
            cntlin  = inds;
run;
```

Out of the 21 **CNTLIN** columns, 5 will adequately describe the format for our purpose

| FMTNAME | START | LABEL | TYPE | HLO |
|---|---|---|---|---|
| COHORT | 12345-001-001 | A | C | |
| COHORT | 12345-001-002 | A | C | |
| COHORT | 12345-001-003 | B | C | |
| COHORT | 12345-001-004 | B | C | |

*The above example shows how the previous format (COHORT) would be stored as a dataset.*

# The FMTMERGE Macro

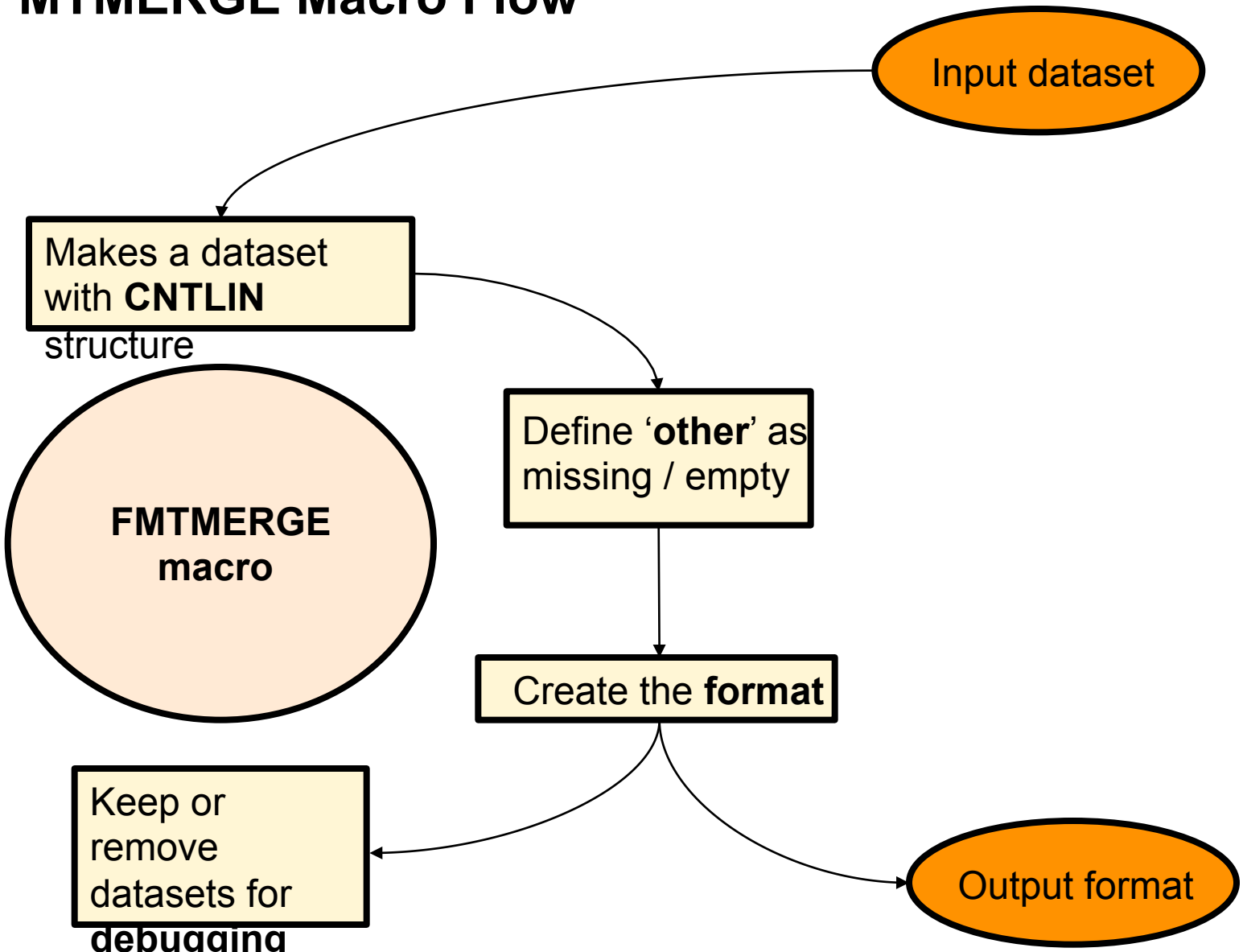
| Keyword Parameter | Description |
|---|---|
| **FNAME** | Name of input dataset and variable containing the formatted value. Will also be the name of the output format. |
| **FSTART** | Variable containing the unformatted data value e.g. USUBJID *[default]* |
| **FTYPE** | C = Character format *[default]*<br>N = Numeric format<br>I = Numeric informat<br>J = Character informat |
| **FDEBUG** | Y – keeps the output dataset *[default]*<br>N – deletes the output dataset |

**Examples:-**

1) `%fmtmerge(fname=RACE)`

2) `%fmtmerge(fname=WEIGHT, fstart=PT, ftype=N, fdebug=N)`

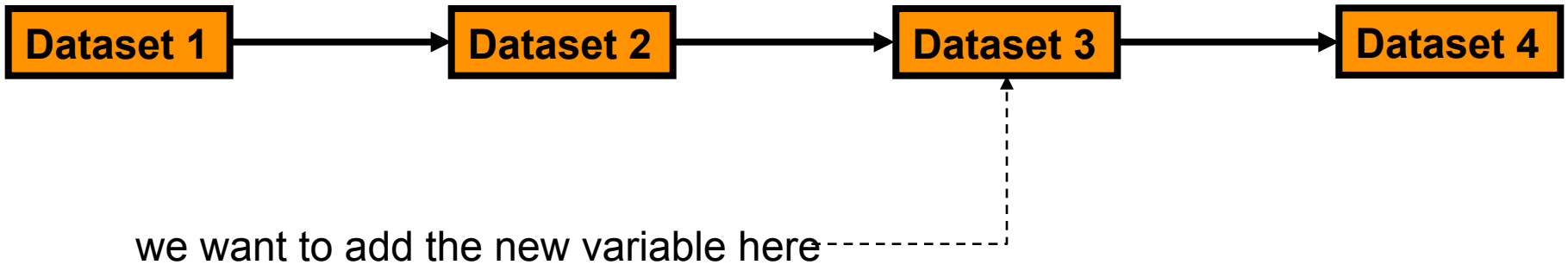
# FMTMERGE Macro Flow



# Example – Dataset Maintenance

Program code already exists and we have been asked to add in a new variable which can be added as a 1 to 1 merge e.g. a patient level variable.

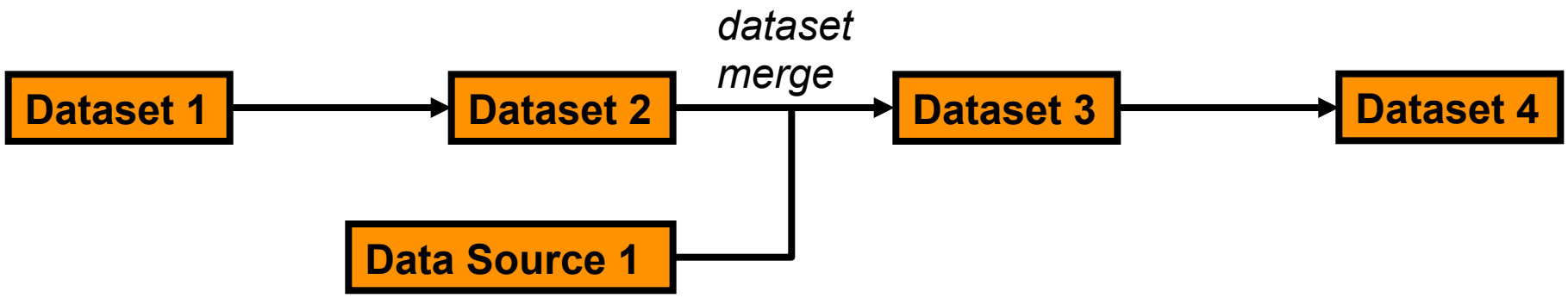
Here is a graphical representation of the program flow:-



# Example – Dataset Maintenance

We could make a new dataset in the middle of the existing code which has our required new variable and join it onto Dataset 2.

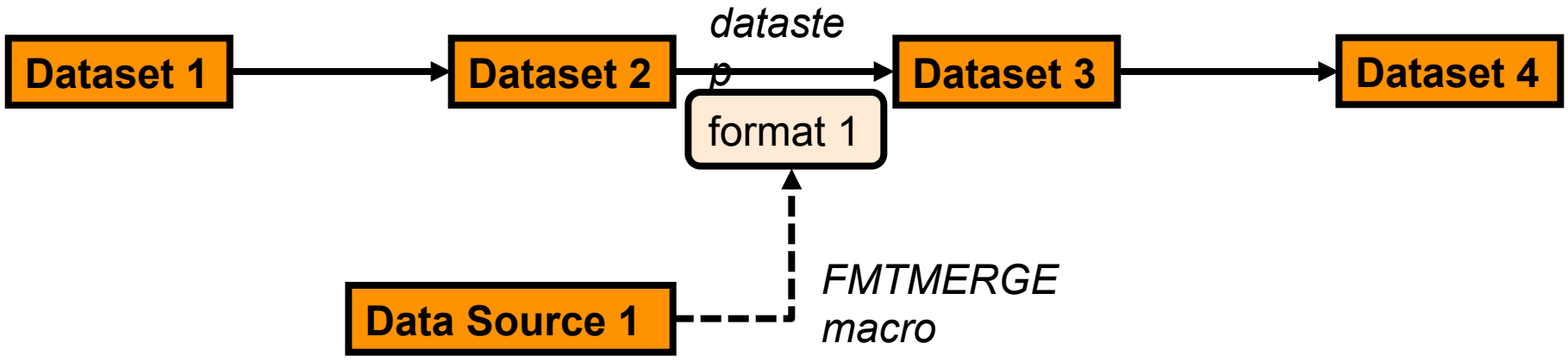
A graphical representation of the program flow is shown below:-



# Example – Dataset Maintenance

Using the FMTMERGE macro we can process the data outside of the original program flow, store it as a format and apply in an already existing data step.

A graphical representation of the program flow is shown below:-



# Summary

- ☺ Formats allow a fast and clean way to join data
- ☺ The CNTLIN option in PROC FORMAT allows formats to be created dynamically
- ☺ The FMTMERGE macro provides a simple way to create data driven formats
- ☺ Data driven formats are a useful additional technique available to the SAS programmer

# Acknowledgements


☺ Vincent Buchheit

☺ Johann Laurent

☺ Timothy Barnett

☺ Philip Holland

☺ Jim Elder

# *Doing now what patients need next*

# Backup Slides

# FMTMERGE Macro – Full Code

```
 options mprint mlogic;


%macro fmtmerge(fname  = ,
                ftype  = C,
                fstart = usubjid,
                fdebug = Y);


  %* make the dataset which stores the values for the format *;
  data &fname.dpre (keep = fmtname type start label);
    set &fname;
    fmtname = "&fname";
    type = "&ftype";
    start = &fstart;
    rename &fname = label;
  run;


  %* create 'other' values *;
  data other;
    type    = "&ftype";
    hlo     = "O";
    fmtname = "&fname";
  run;
```

```
%* combine both datasets *;
  data &fname.d;
    set &fname.dpre
        other;
  run;


  %* create the format *;
  proc format library=work.formats
cntlin=&fname.d;
  run;


  %* remove temporary dataset *;
  proc datasets nodetails nolist;
    delete &fname.dpre other;
  run;
  quit;


  %* remove format dataset - upon user request *;
  %let fdebugu = %upcase(&fdebug);
  %if &fdebugu=N %then %do;
    proc datasets nodetails nolist;
      delete &fname.d;
    run;
              quit;
  %end;


%mend fmtmerge;
```

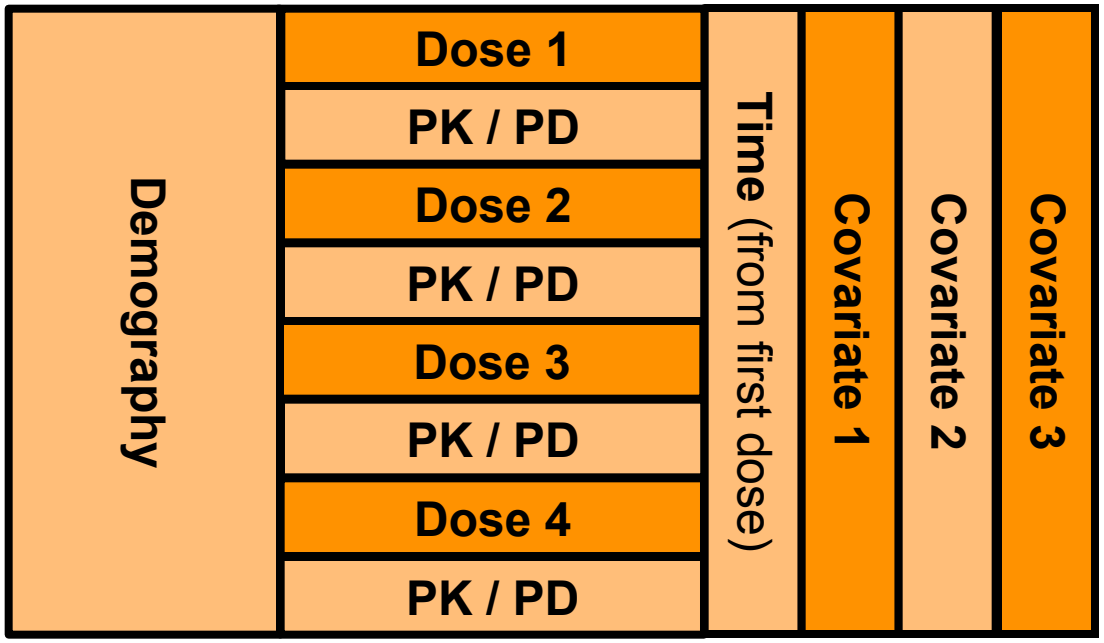
# Creating the Macro

The CNTLIN structure can be used as a basis for the data driven formats macro. To make the data driven formats macro some macro parameters will be defined:

| CNTLIN Column | Description | Macro Parameter Name / Usage |
|---|---|---|
| FMTNAME | *The name of our format* | FNAME |
| START | *The unformatted value of our format* | FSTART |
| LABEL | *The formatted value of our format* | FNAME |
| TYPE | *C = Character format*<br>*N = Numeric format*<br>*I = Numeric informat*<br>*J = Character informat* | FTYPE |
| HLO | *Range information* | Will use to hold value for 'other' as missing / empty |

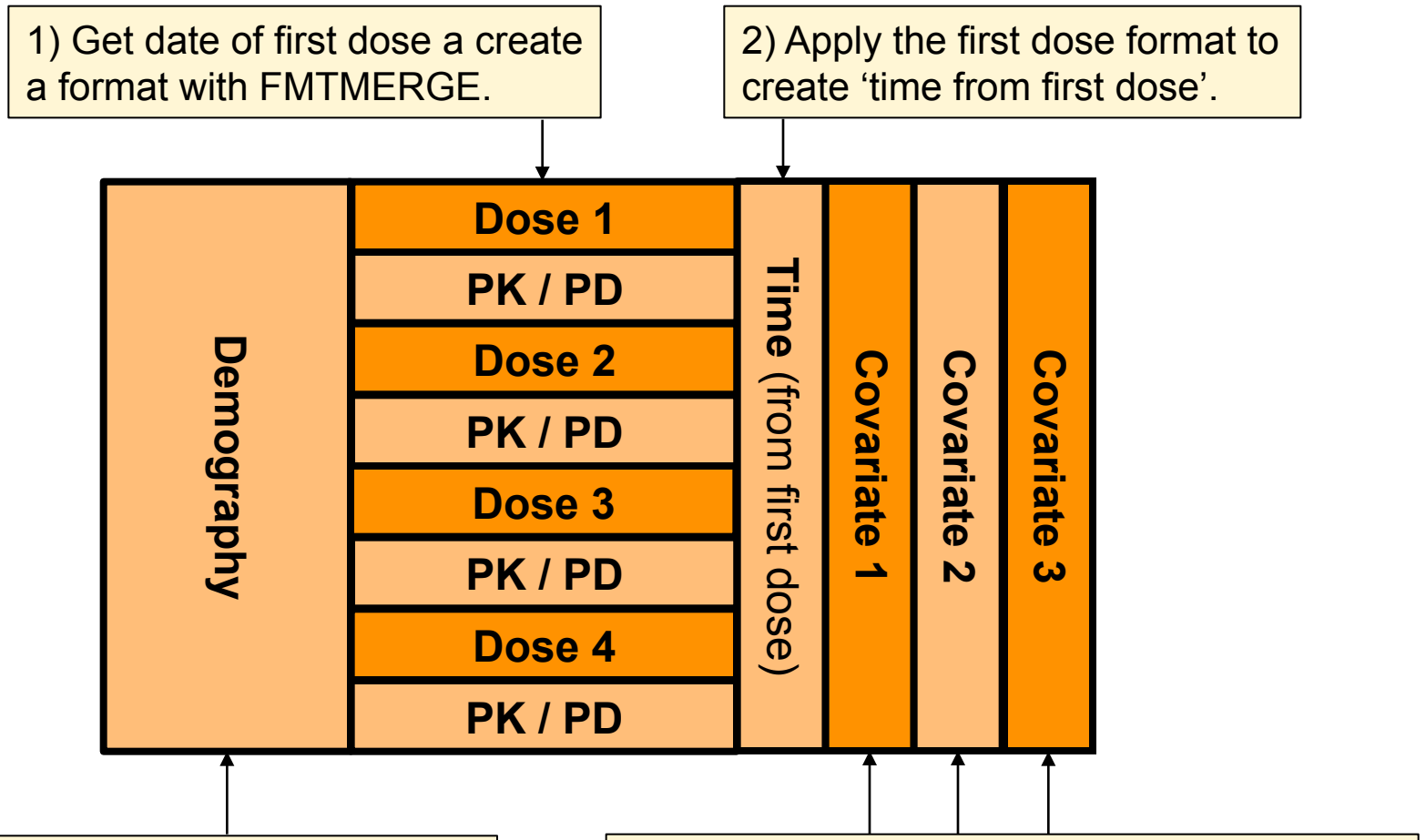
# Example – Analysis Dataset

Here is a graphical example of an analysis / modelling dataset:




Let's look at how we can use the FMTMERGE macro in it's construction.

# Example – Analysis Dataset



# Using the FMTMERGE Macro to Create a Format

**1)** Make an input dataset e.g:

| USUBJID | COHORT |
|---|---|
| 12345-001-001 | A |
| 12345-001-002 | A |
| 12345-001-003 | B |
| 12345-001-001 | B |

dataset name = COHORT

**2)** Run the macro:

```
%fmtmerge(fname = cohort);
```

to get a character format called '*cohort*'

**3)** Apply the format in the code to add the '*cohort*' column to your data:

```
put(usubjid, $cohort.);
```

The macro has defaults for **FTYPE** = C and **FSTART** = USUBJID.

*Note: The dataset name, the column holding the formatted value and the output format name are the same.*

# CNTLIN Dataset

The **CNTLIN** option in **PROC FORMAT** allows creation of a format from a dataset

```
proc format library = work.formats
            cntlin  = inds;
run;
```

Out of the 21 **CNTLIN** columns, 5 will adequately describe the format for our purpose

| Column | Description |
|---|---|
| FMTNAME | *The name of our format* |
| START | *The unformatted value of our format* |
| LABEL | *The formatted value of our format* |
| TYPE | *e.g. Character, Numeric, Format, Informat, Picture* |
| HLO | *Range information* |

# *Doing now what patients need next*