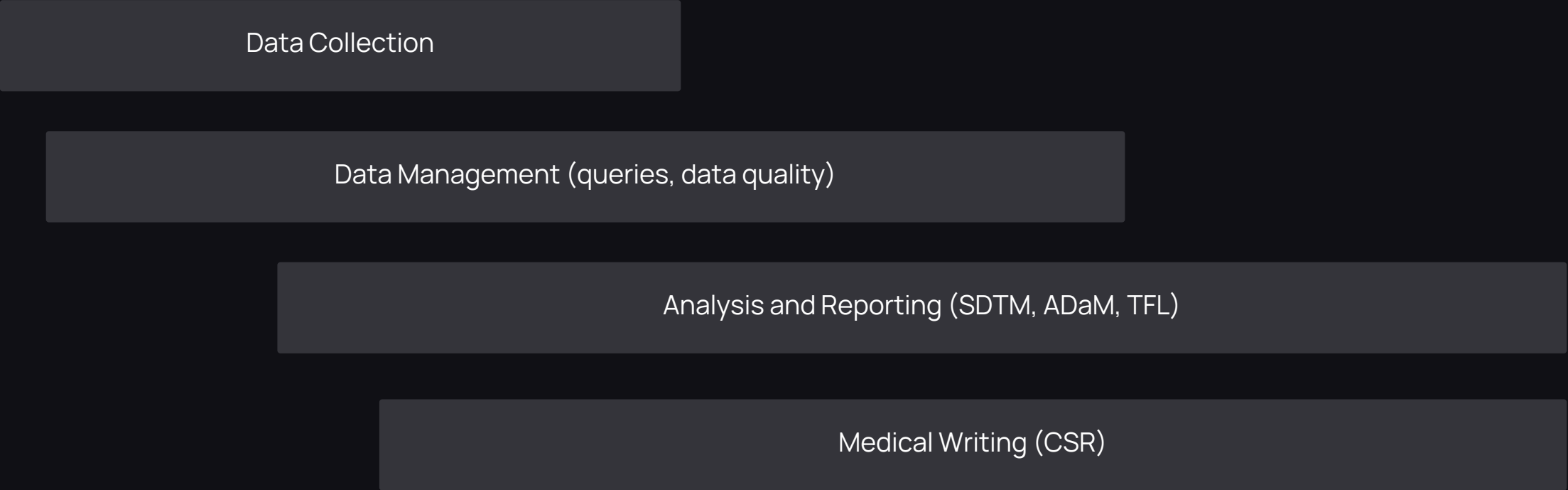




Increasing Submission Speed and Quality using Traceability and AI

Drug submission data workflow



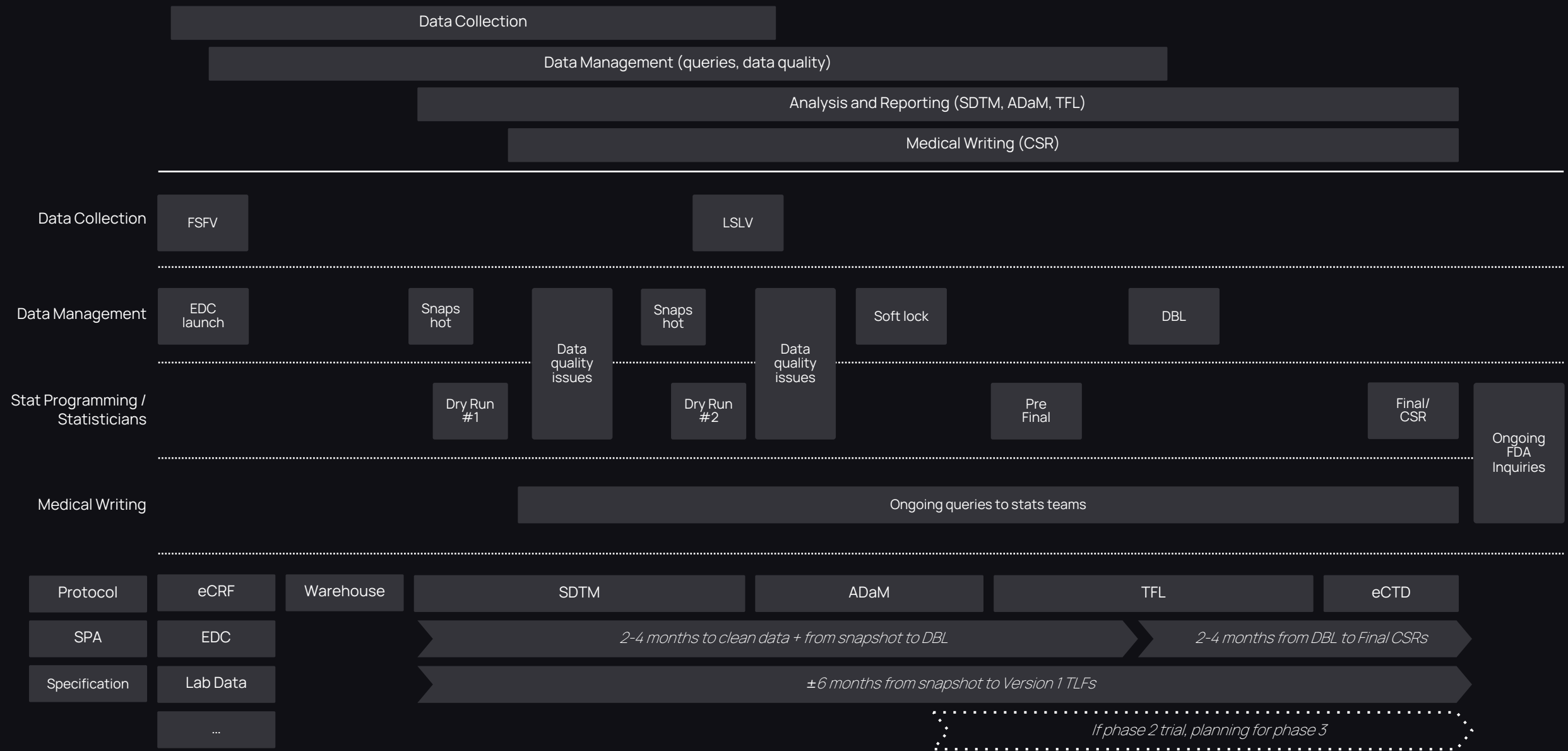
Data Collection

Data Management (queries, data quality)

Analysis and Reporting (SDTM, ADaM, TFL)

Medical Writing (CSR)

Drug submission data workflow



Understanding and Communicating Studies is Difficult

Programming teams

Statisticians

Medical Writers

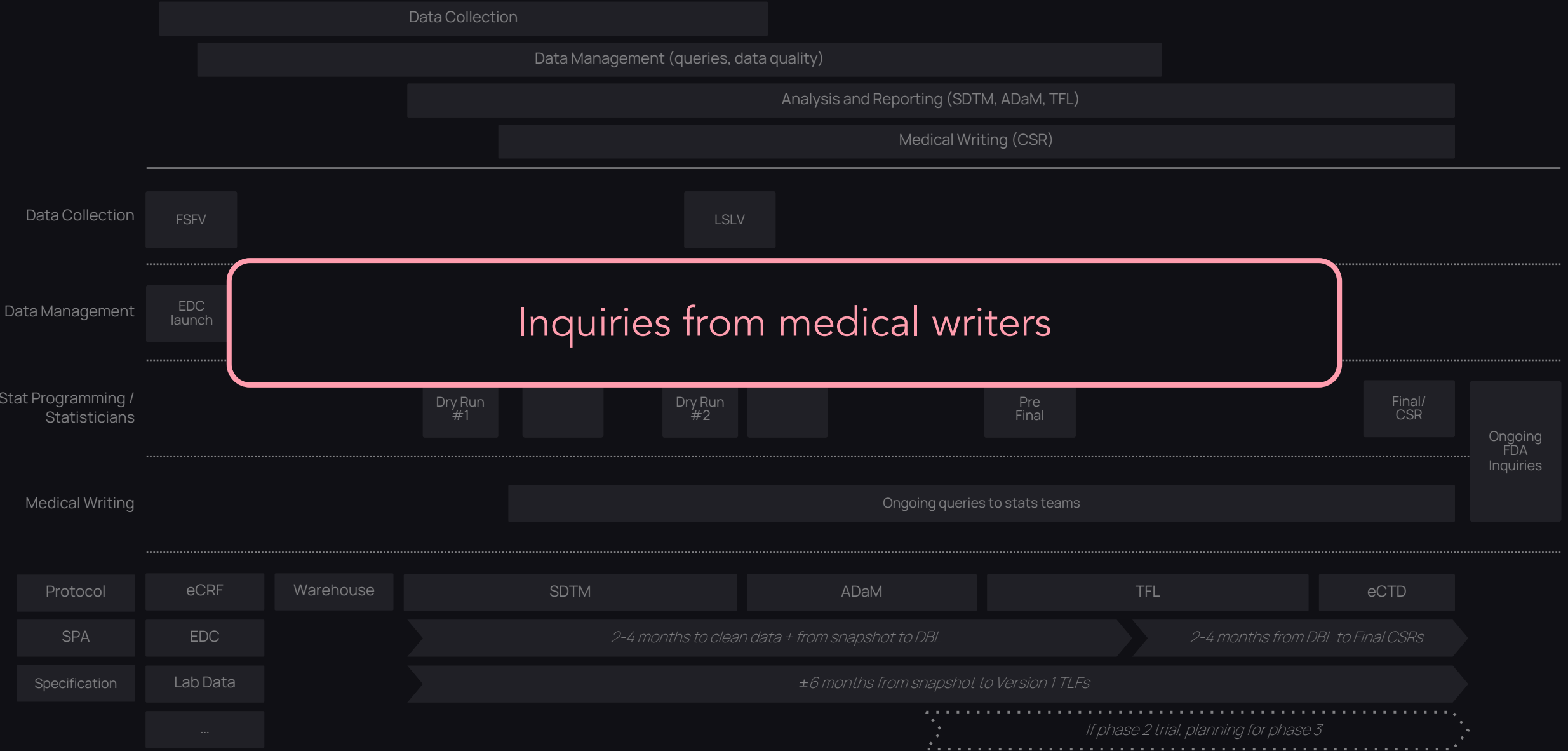
Medical Experts

Internal Reviewers

Regulatory Authorities

Medical Affairs

Drug submission data workflow



Inquiry

Why is the # of adverse events reported much higher than anticipated?

Inquiry

Why is the # of adverse events reported much higher than anticipated?



Hypothesis

Is the definition of treatment-emergent time-frame correctly implemented?

Inquiry

Why is the # of adverse events reported much higher than anticipated?



Hypothesis

Is the definition of treatment-emergent time-frame correctly implemented?



Root Cause Analysis

Find code

Find
implementation

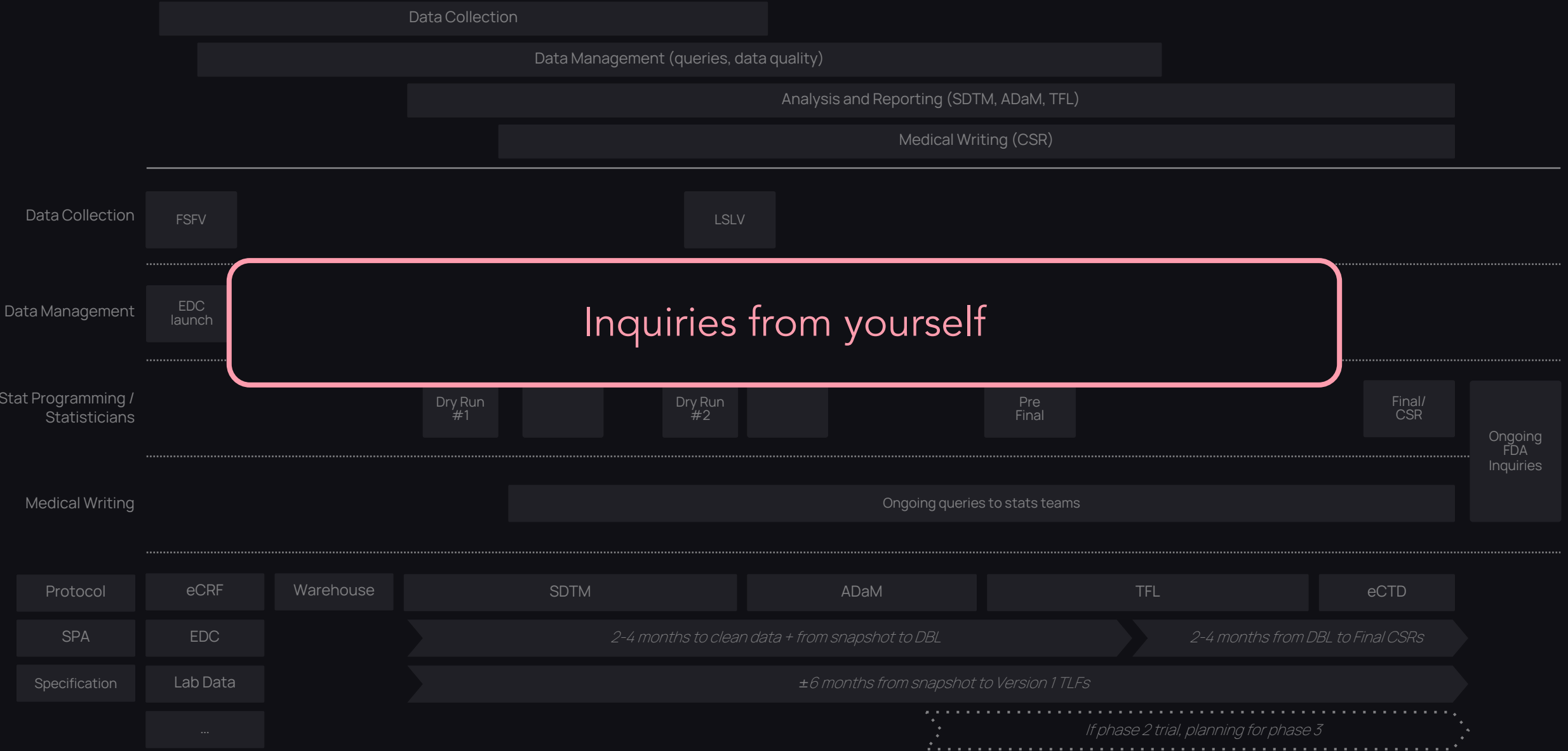
Find macros

Find logs

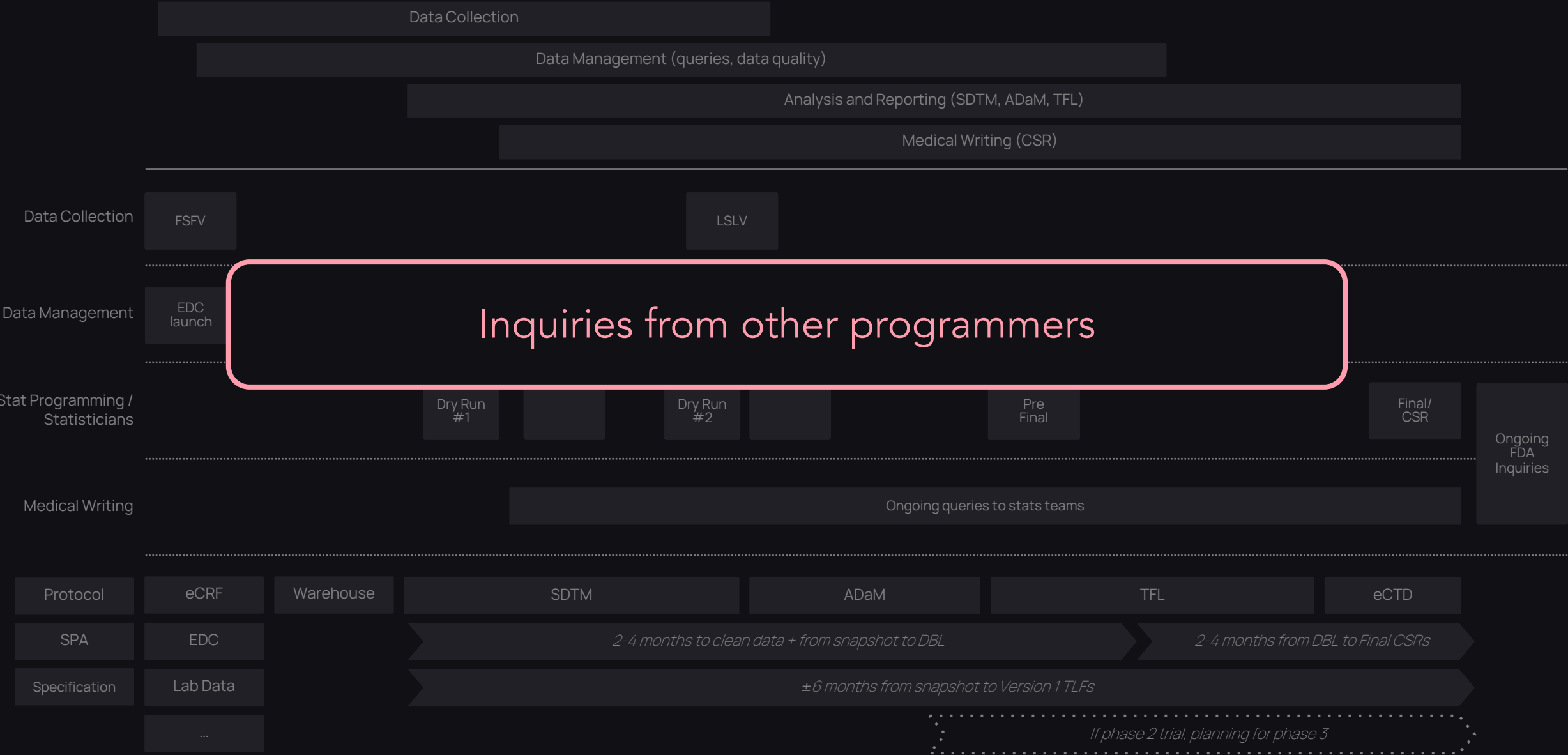
Resolve
macros

...

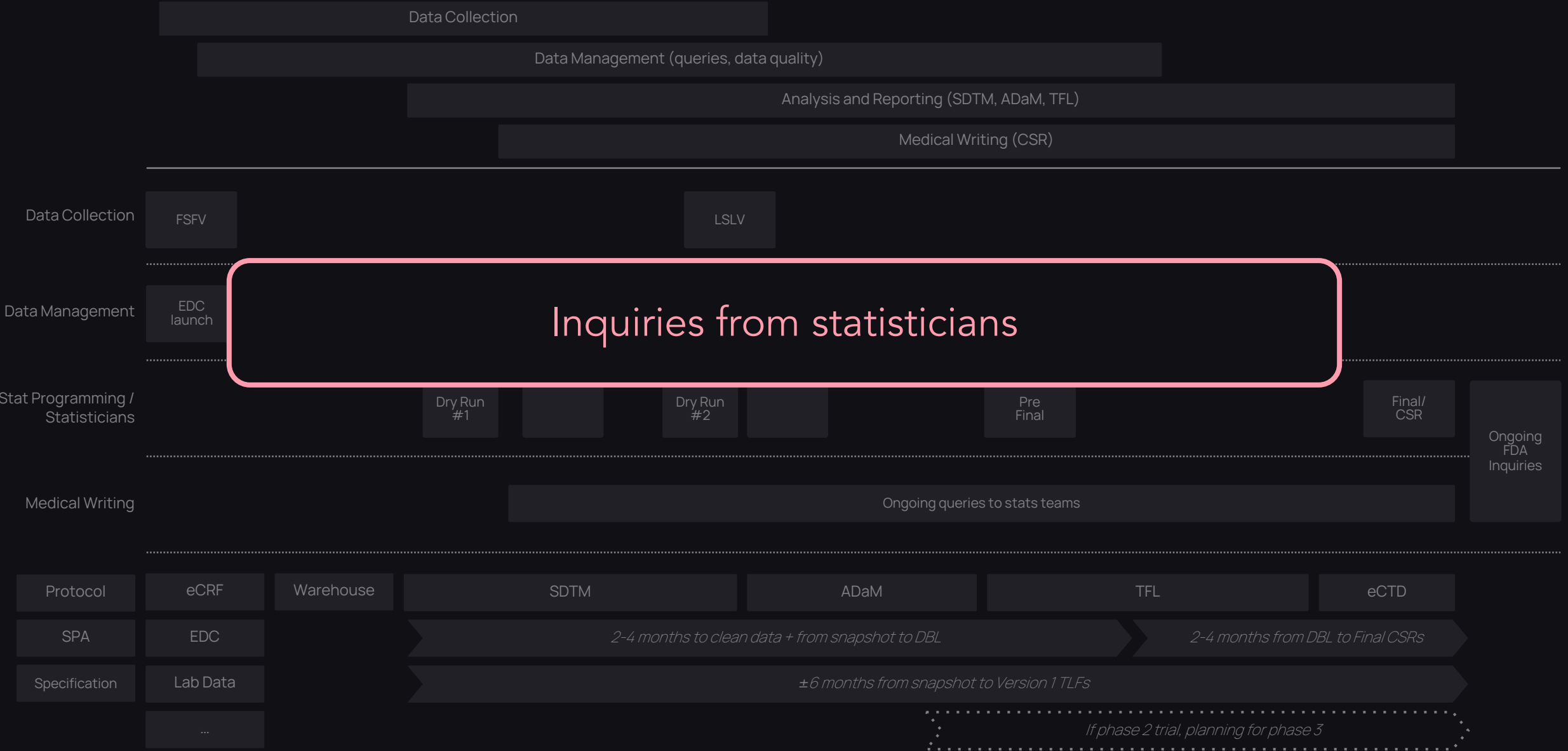
Drug submission data workflow



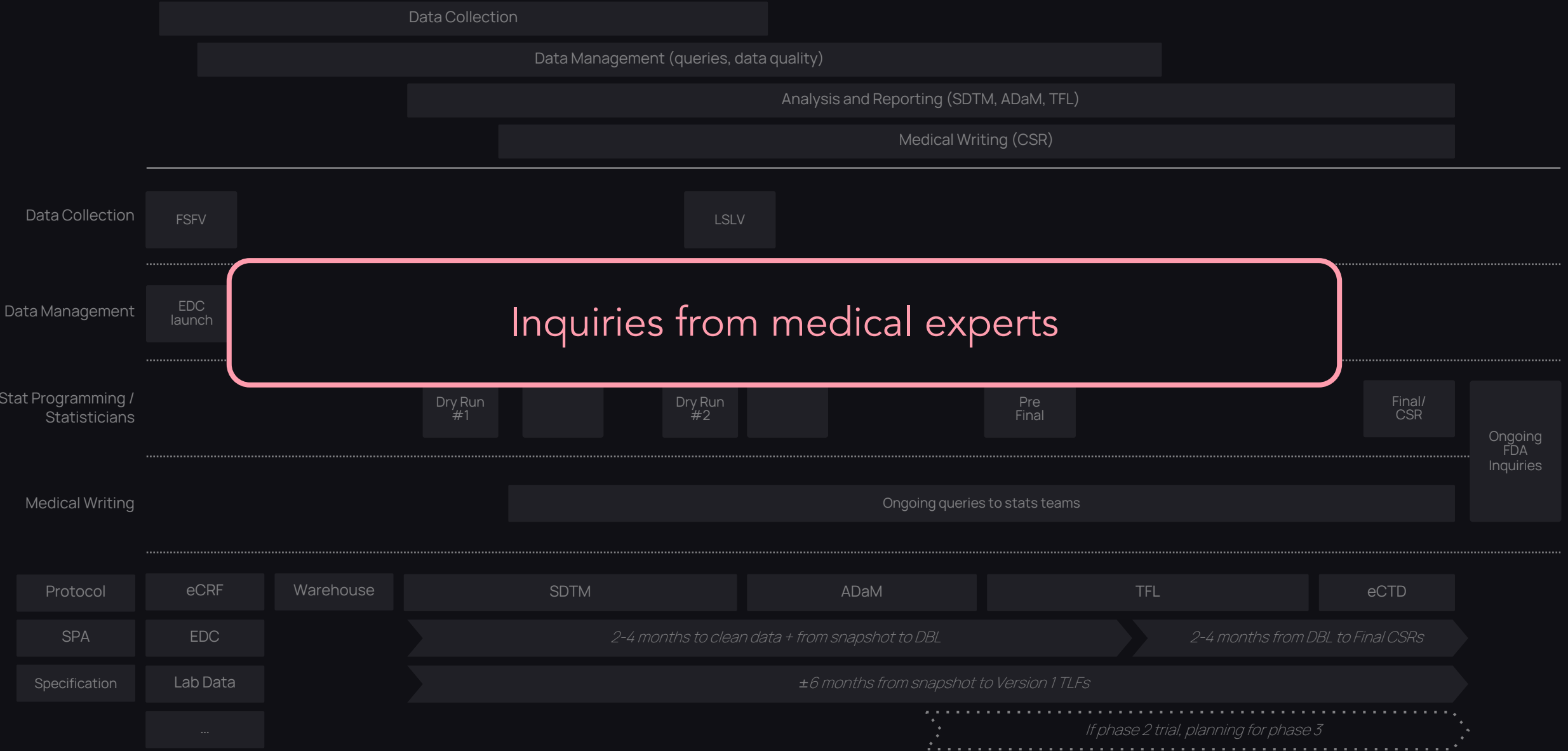
Drug submission data workflow



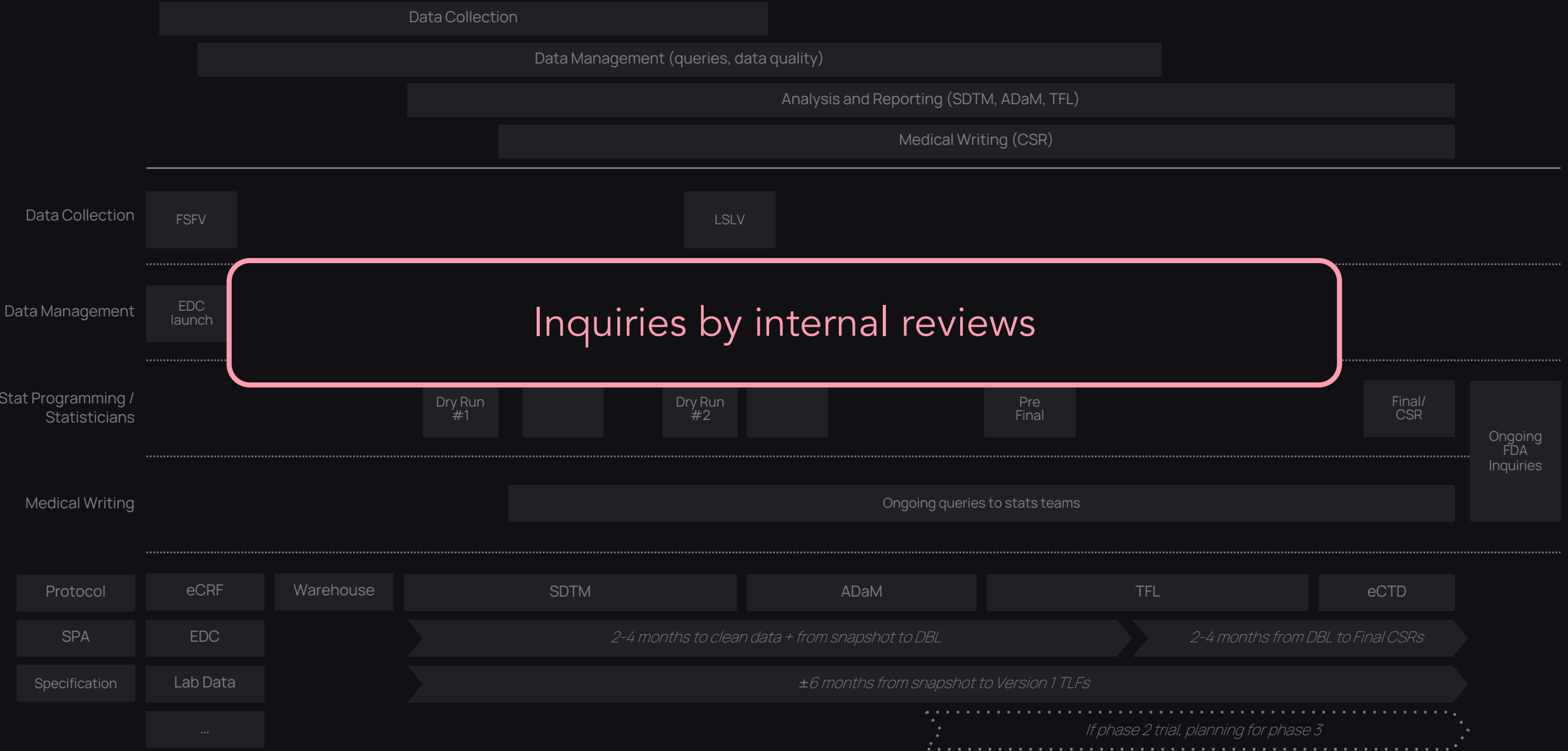
Drug submission data workflow



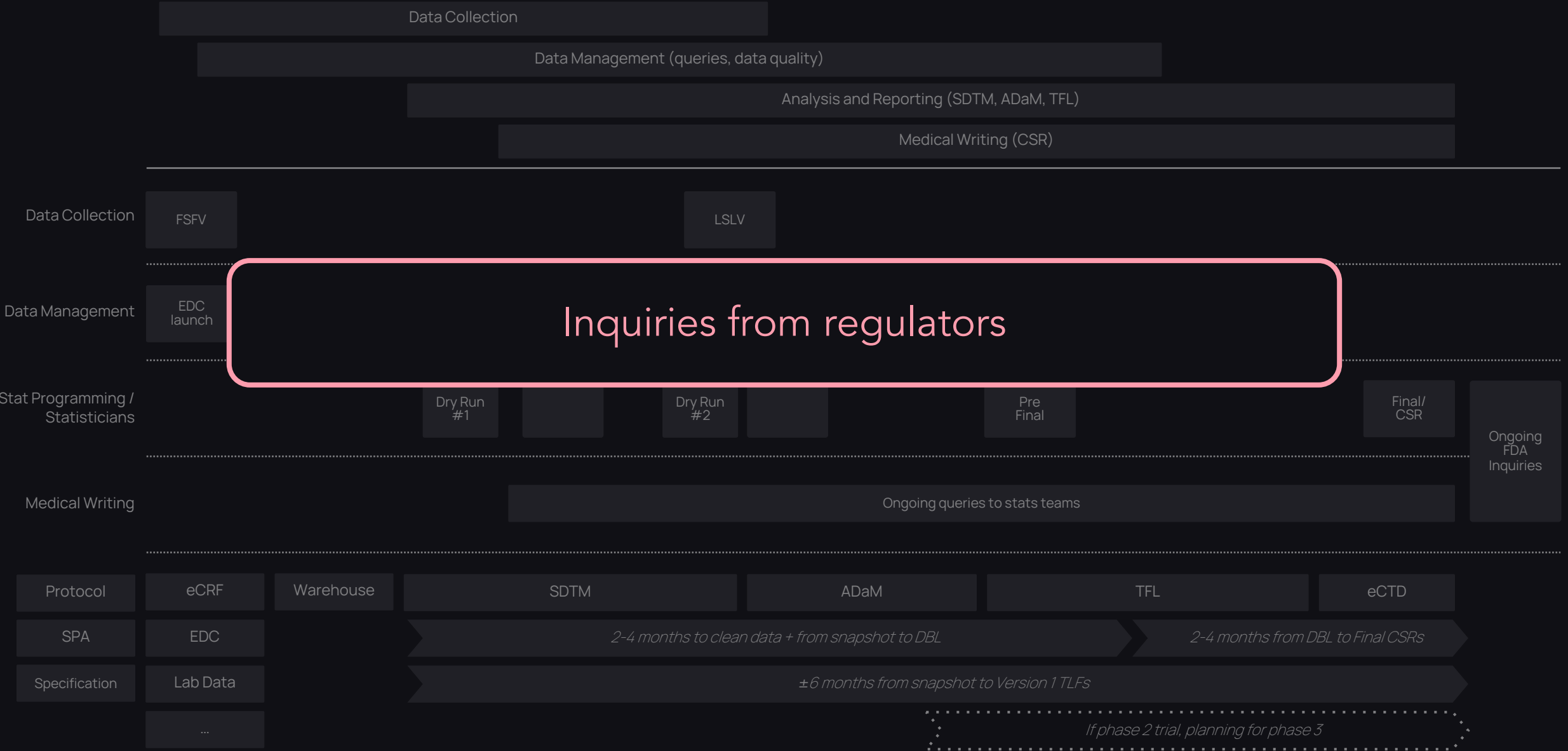
Drug submission data workflow



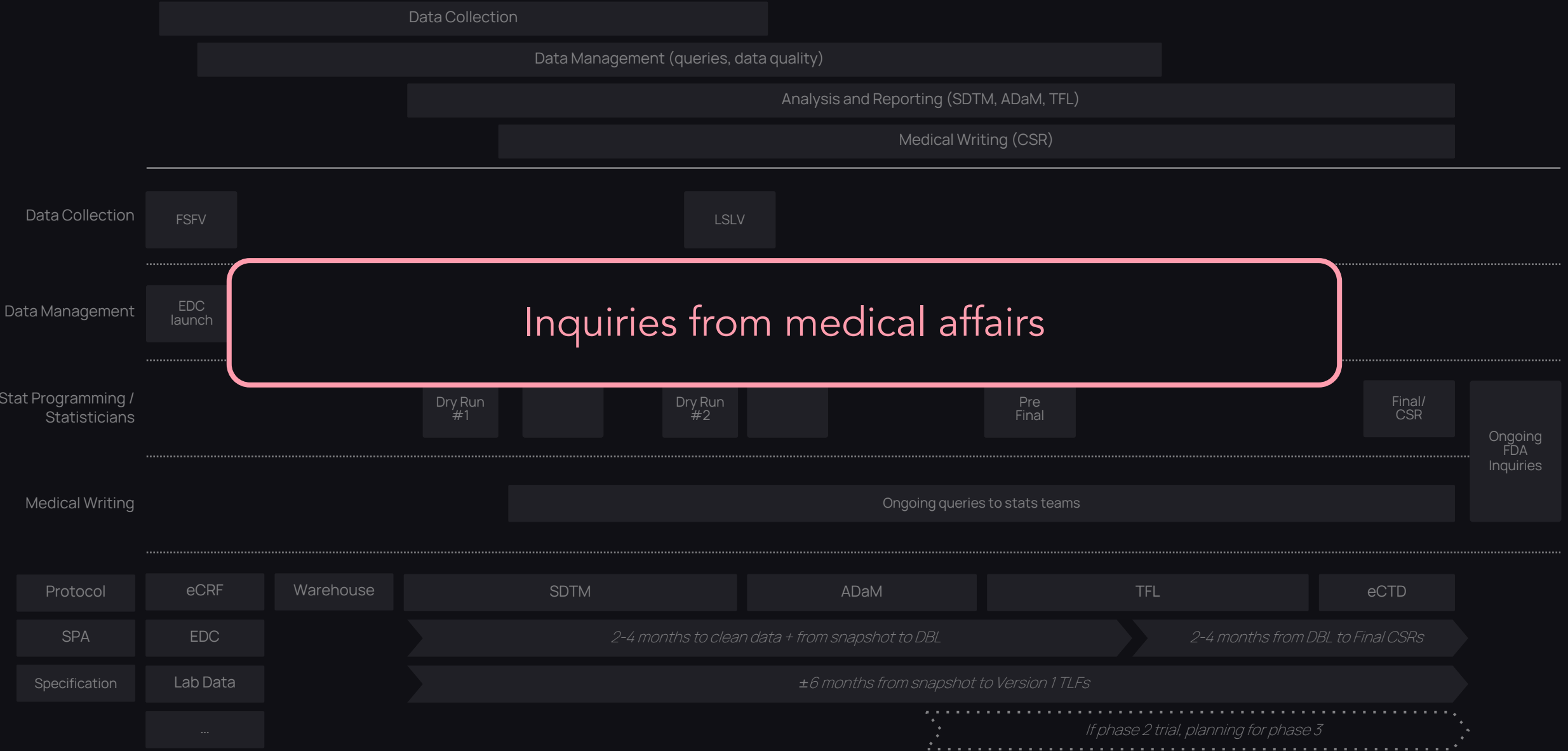
Drug submission data workflow



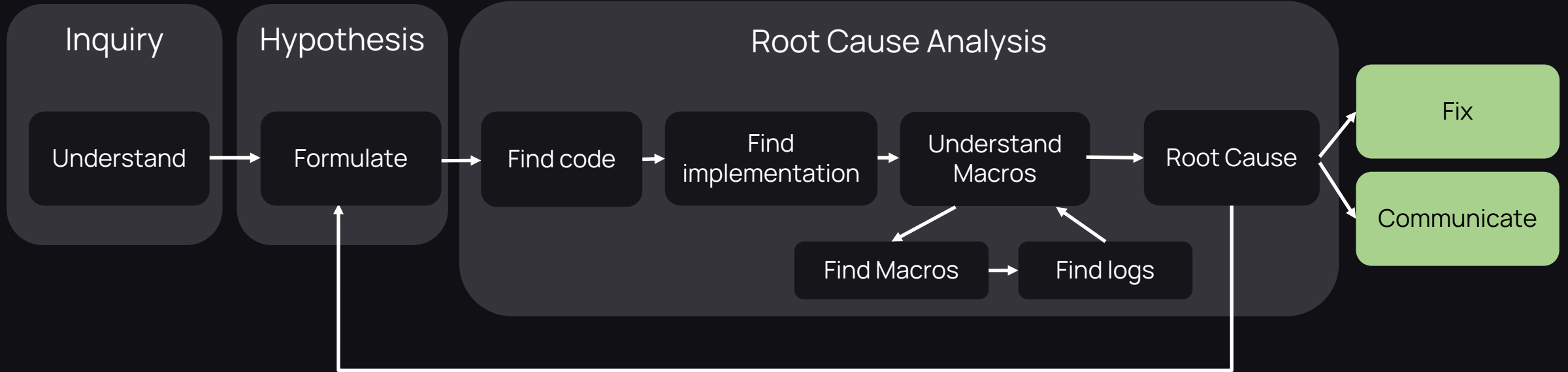
Drug submission data workflow



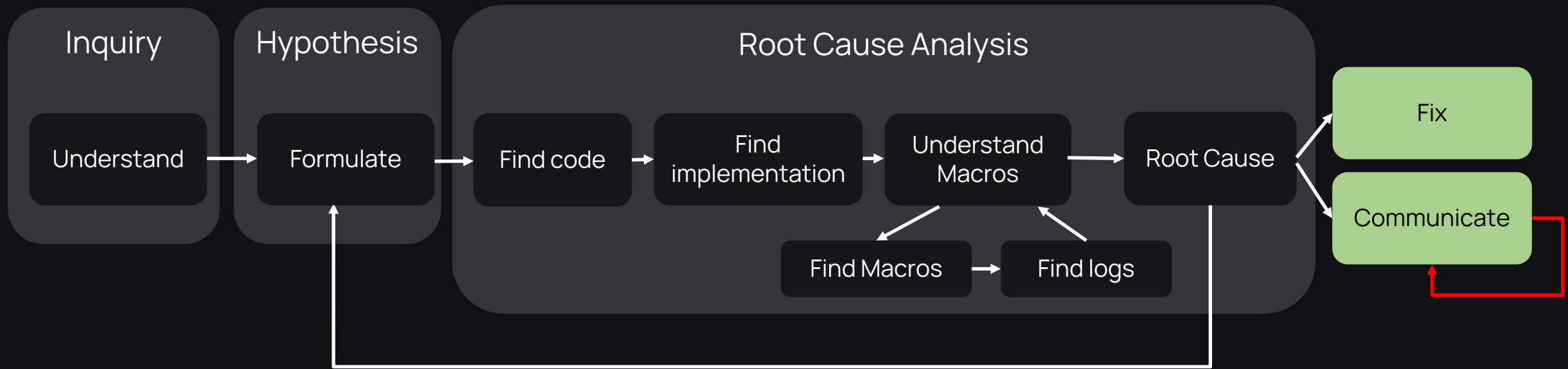
Drug submission data workflow



Inquiry Response Cycle



Inquiry Response Cycle

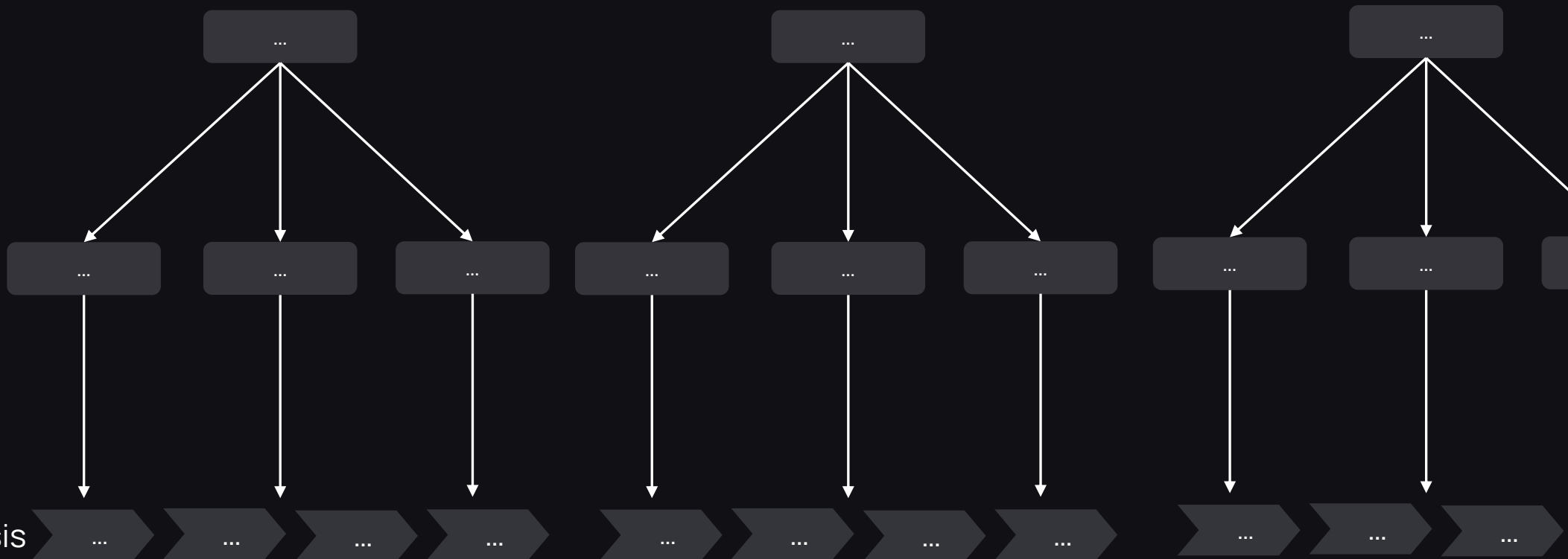


Not just one question...

Inquiries

Hypotheses

Root Cause Analysis



Communicating and Understanding Studies is Difficult

Quality is **less** than what can be achieved

Programmers

- Immense workload
- Deadline pressure
- Low value work

Sponsors

- Slow review and inquiry-response cycles
- Slow iterations
- Little explorative work

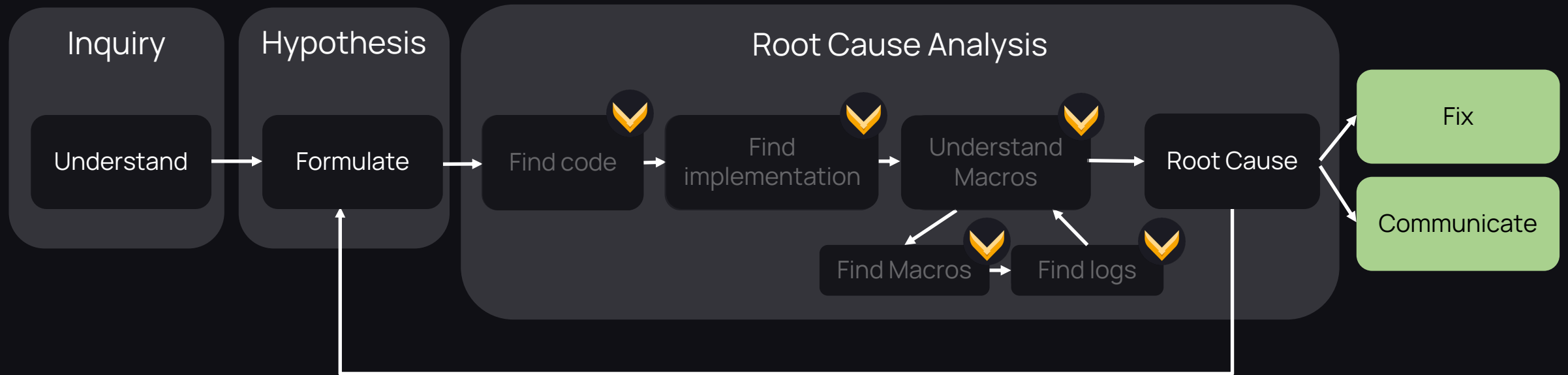
Regulators

- Difficult to understand studies
- Slow review and inquiry-response cycles

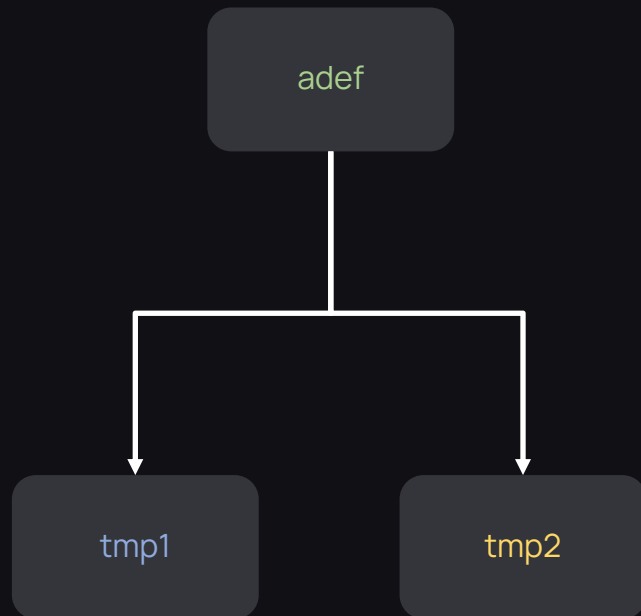


Build, Validate and Submit Clinical Trial Analyses

Proven reduced inquiry-response turn-around time

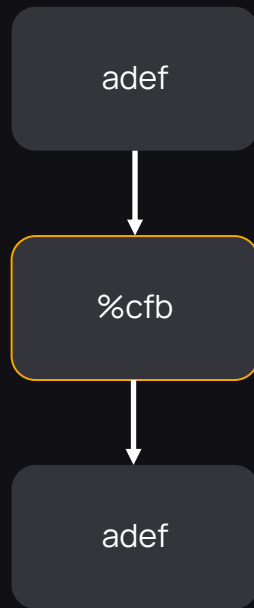


Full study traceability



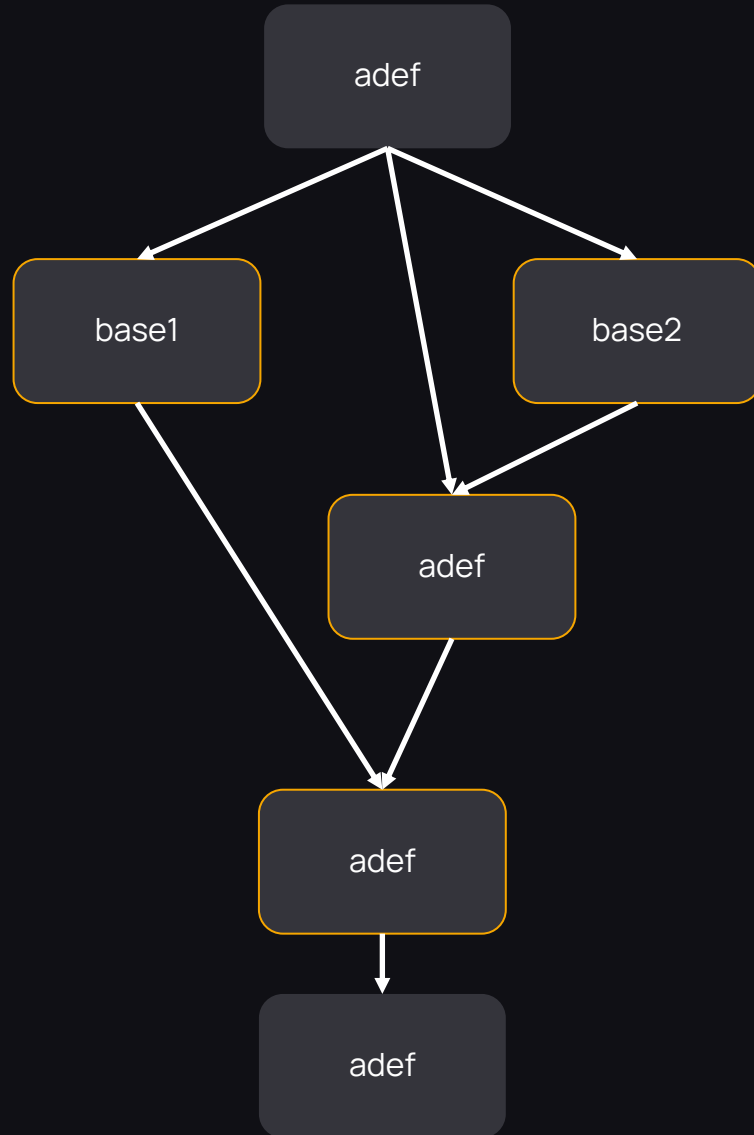
```
data ade;  
    ...  
run;  
  
data tmp1;  
    set ade;  
    ...;  
run;  
  
data tmp2;  
    set ade;  
    ...;  
run;
```

Macro resolution



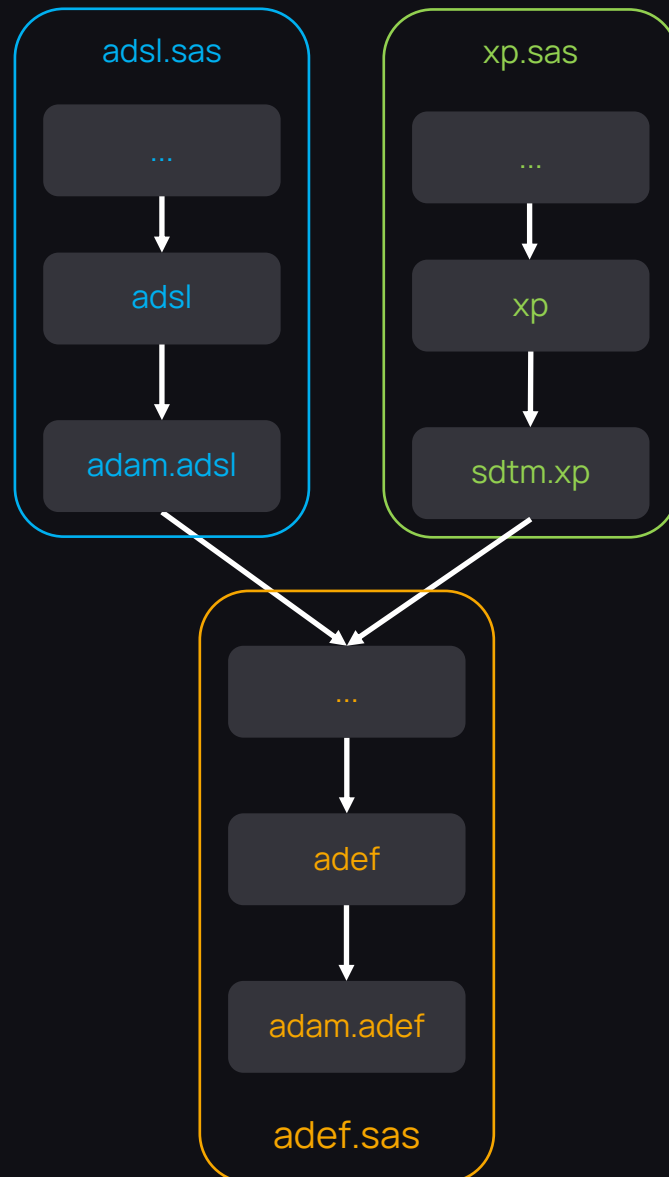
```
data adef;  
    ...  
run;  
  
%cfb(indata=adef, outdata=adef);  
  
data adef;  
    set adef;  
    ...  
run;
```

Macro resolution



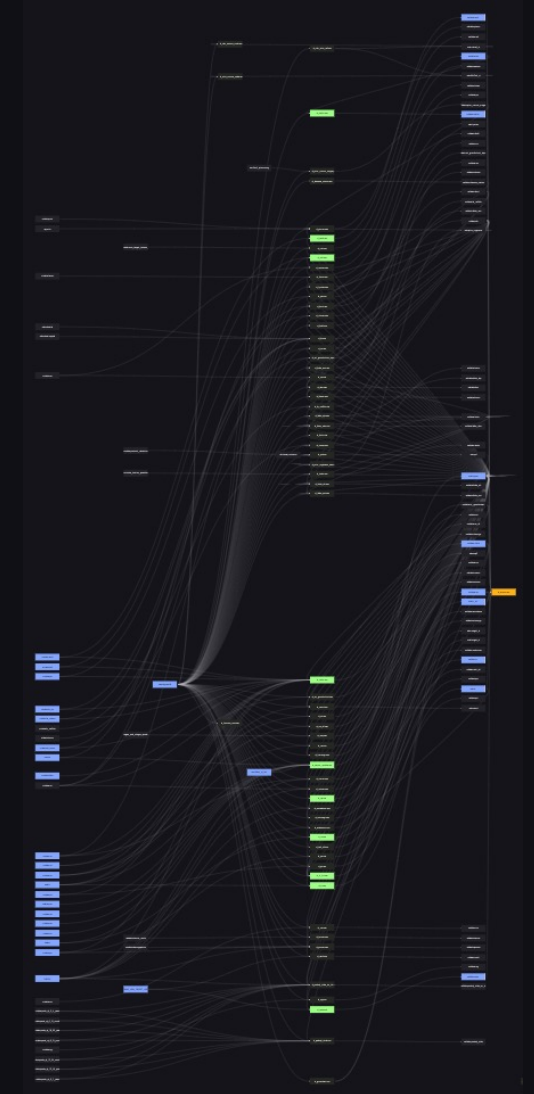
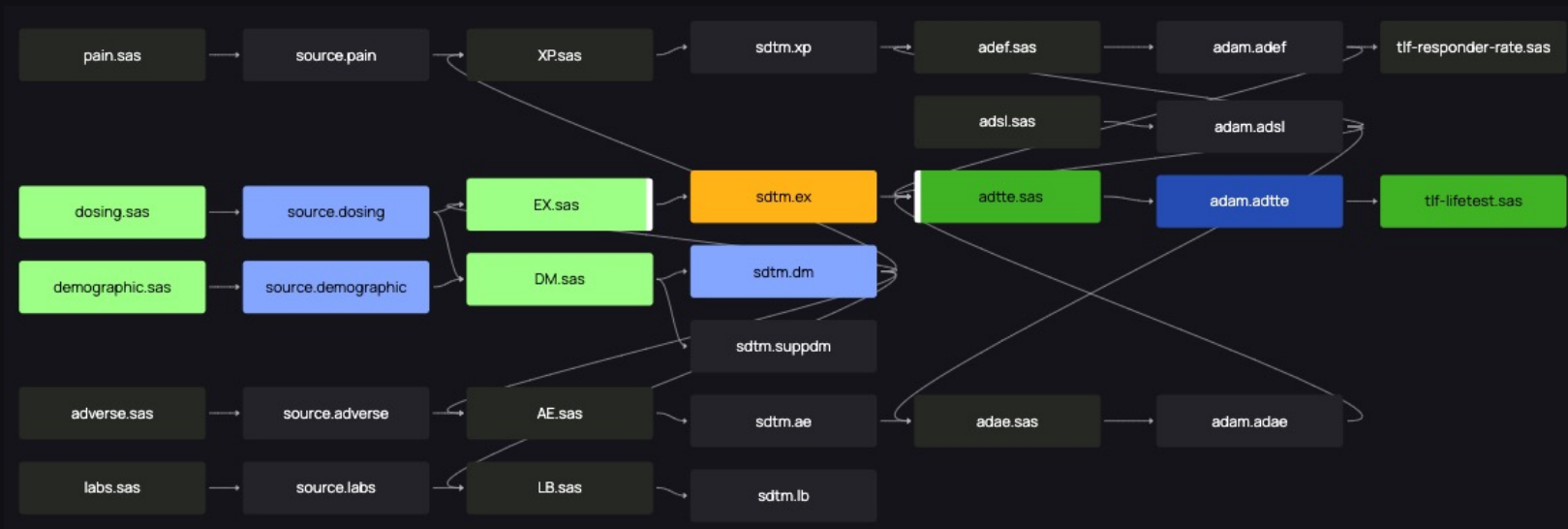
```
data adef;  
  ...  
run;  
  
data base1 base2;  
  set adef;  
  ...  
run;  
  
data adef;  
  merge adef base2;  
  by usubjid visitnum;  
  ...  
run;  
  
data adef;  
  merge base1 adef;  
  by usubjid;  
  ...  
run;  
  
...
```

Derivation



```
...  
...  
...  
  
data adam.adsl;  
    set adsl;  
    ...  
run;  
  
...  
...  
...  
  
data sdtm.xp;  
    set adsl;  
    ...  
run;  
  
...  
...  
...  
  
data adam.adeft;  
    set adef;  
    by usubjid visitnum;  
    ...  
run;
```

Overview **Graph** of a Study



Focus on upstream **dependencies** and downstream **outputs**

Select files

ad

Files

Study

adae.sas

adef.sas

adsl.sas

adtte.sas

adverse.sas

Dataset Graph

Overview Graph

File: adtte.sas

adam.adae

adae

adae

adam.adsl

adtte

EMPTY_ADTTE

adtte

adam.adtte

adam.adef

adef

adef

sdm.ex

lstex

lstex

+

React Flow

Original

Resolved

Log

Derivation

adtte.sas (Original)

```
98      cnsr = 2;
99      adt = asdt;
100     evntdesc = put(cnsr, evntdesc.) ;
101     srcdom = 'ADAE';
102     srcvar = 'ASTDY';
103     srcseq = aeseq;
104   end;
105   else
106   do;
107     cnsr = 3;
108     adt = trtedt;
109     evntdesc = put(cnsr, evntdesc.) ;
110     srcdom = 'ADSL';
111     srcvar = 'TRTEDT';
112     srcseq = .;
113   end;
114
115   aval = adt - startdt + 1;
116
117   format startdt adt yymmdd10.;
118   run;
119
120   ** assign variable order and labels;
121   data adtte;
122     retain &adtteKEEPSTRING;
123     set EMPTY_adtte adtte;
124   run;
125
126   **** SORT adtte ACCORDING TO METADATA AND SAVE PERMANENT DATASET;
127   %make_sort_order(metadatafile=&metadatafile, dataset=ADTTTE)
128
```

Log Messages

Type

Log message

warning 1

WARNING: Multiple lengths were specified for the variable AGEGR1 by input data set(s). This can cause truncation of data.

Inspect original Logs

Original Resolved Log Derivation

adef.sas (Original)

```
13 out = adsl;
14 by usubjid;
15
16 data adef;
17 merge sdtm.xp (in = inadef) adsl (in = inadsl);
18 by usubjid;
19
20 if not(inadsl and inadef) then
21   put 'PROB' 'LEM: Missing subject?-- ' usubjid= inadef= inadsl;
22
23   rename trt01p = trtp
24         trt01pn = trtpn
25         xptest = param
26         xptestcd = paramcd
27         visit = avisit
28         xporres = avalc
29   ;
30   if inadsl and inadef;
31   avisitn = input(put(visitnum, avisitn.), best.);
32
33   %dtc2dt(xpdtc, refdt=randdt);
34 run;
35
36 ** calculate changes from baseline for all post-baseline visits;
37 %cfb(indata=adef, outdata=adef, dayvar=ady, avalvar=xpstresn);
38
39 data adef;
40 set adef;
41 by usubjid;
42
43 retain crit1 "Pain improvement from baseline of at least 2 points";
44 RESPFL = put((.z <= chg <= -2), _0n1y.);
45 if RESPFL='Y' then
46   crit1fl = 'Y';
47 else
48   crit1fl = 'N';
49 run;
50
51 ** assign variable order and labels;
52 data adef;
53 retain &ADEFKEEPSTRING;
54 set EMPTY_ADEF adef;
55 run;
56
57 **** SORT ADEF ACCORDING TO METADATA AND SAVE PERMANENT DATASET;
58 %make_sort_order(metadatafile=&metadatafile, dataset=ADEF)
59
60 proc sort
61   data=adef(keep = &ADEFKEEPSTRING)
```

Original Resolved Log Derivation

_run_all.log (Showing lines relating to adef.sas)

```
-----
6384 MLOGIC(CFB): Parameter OUTDATA has value adef
6385 MLOGIC(CFB): Parameter DAYVAR has value ady
6386 MLOGIC(CFB): Parameter AVALVAR has value xpstresn
6387 MLOGIC(CFB): Parameter KEEPVAR has value
6388 MPRINT(CFB): proc sort data = adef out = adef (rename = (xpstresn = aval));
6389 MPRINT(CFB): by usubjid visitnum;
6390 MPRINT(CFB): run;
6391
6392 NOTE: There were 177 observations read from the data set WORK.ADEF.
6393 NOTE: SAS sort was used.
6394 NOTE: The data set WORK.ADEF has 177 observations and 30 variables.
6395 NOTE: PROCEDURE SORT used (Total process time):
6396     real time           0.00 seconds
6397     cpu time            0.01 seconds
6398
6399
6400 MPRINT(CFB): * Baseline is defined as the last non-missing value prior to study day 1 first dose;
6401 MPRINT(CFB): * (note, values on Day 1 are assumed to occur before the first dose);
6402 MPRINT(CFB): data base1 (keep = usubjid visitnum) base2 (keep = usubjid base);
6403 MPRINT(CFB): set adef;
6404 MPRINT(CFB): where ady<=1 and aval > .z;
6405 MPRINT(CFB): by usubjid visitnum;
6406 MPRINT(CFB): rename aval = base;
6407 MPRINT(CFB): if last.usubjid;
6408 MPRINT(CFB): run;
6409
6410 NOTE: There were 60 observations read from the data set WORK.ADEF.
6411 WHERE (ady<=1) and (aval>.Z);
6412 NOTE: The data set WORK.BASE1 has 60 observations and 2 variables.
6413 NOTE: The data set WORK.BASE2 has 60 observations and 2 variables.
6414 NOTE: DATA statement used (Total process time):
6415     real time           0.00 seconds
6416     cpu time            0.01 seconds
6417
6418
6419 MPRINT(CFB): * Do one merge to identify the baseline record;
6420 MPRINT(CFB): data adef;
6421 MPRINT(CFB): merge adef base1 (in = dbase1);
```

See original and macros resolved

```
Original Resolved Log Derivation
adef.sas (Original)

13 out = adsl;
14 by usubjid;
15
16 data adef;
17 merge sdtm.xp (in = inadef) adsl (in = inadsl);
18 by usubjid ;
19
20 if not(inadsl and inadef) then
21 put 'PROB' 'LEM: Missing subject?-- ' usubjid= inadef= inadsl= ;
22
23 rename trt01p = trtp
24 trt01pn = trtpn
25 xptest = param
26 xptestcd = paramcd
27 visit = avisit
28 xporres = avalc
29 ;
30 if inadsl and inadef;
31 avisitn = input(put(visitnum, avisitn.), best.);
32
33 %dtc2dt(xpdtc, refdt=randdt);
34 run;
35
36 ** calculate changes from baseline for all post-baseline visits;
37 %cfb(indata=adef, outdata=adef, dayvar=ady, avalvar=xpstresn);
38
39 data adef;
40 set adef;
41 by usubjid;
42
43 retain crit1 "Pain improvement from baseline of at least 2 points";
44 RESPFL = put((.z <= chg <= -2), _0n1y.);
45 if RESPFL='Y' then
46 crit1fl = 'Y';
47 else
48 crit1fl = 'N';
49 run;
50
51 ** assign variable order and labels;
52 data adef;
53 retain &ADEFKEEPSTRING;
54 set EMPTY_ADEF adef;
55 run;
56
57 **** SORT ADEF ACCORDING TO METADATA AND SAVE PERMANENT DATASET;
58 %make_sort_order(metadatafile=&metadatafile, dataset=ADEF)
59
60 proc sort
61 data=adef(keep = &ADEFKEEPSTRING)
```

```
Original Resolved Log Derivation
adef.sas (Macro-resolved)

15 /
158 run;
159
160 ** calculate changes from baseline for all post-baseline visits;
161 ***** verisian: macro invocation start of <%cfb> *****;
162 *;
163 * %cfb(indata=adef, outdata=adef, dayvar=ady, avalvar=xpstresn);
164 *;
165 *****;
166
167 proc sort data = adef out = adef (rename = (xpstresn = aval));
168 by usubjid visitnum;
169 run;
170
171 * Baseline is defined as the last non-missing value prior to study day 1 first dose;
172 * (note, values on Day 1 are assumed to occur before the first dose);
173 data base1 (keep = usubjid visitnum) base2 (keep = usubjid base);
174 set adef;
175 where ady<=1 and aval > .z;
176 by usubjid visitnum;
177 rename aval = base;
178 if last.usubjid;
179 run;
180
181 * Do one merge to identify the baseline record;
182 data adef;
183 merge adef base1 (in = inbase);
184 by usubjid visitnum;
185 if inbase then
186 ablf1 = 'Y';
187
188 run;
189
190 * Do another merge to merge in the baseline value;
191 data adef;
192 merge adef base2;
193 by usubjid;
194 ;
195 if base > .z then
196 chg = aval - base;
197
198 if base>.z and base ne 0 then
199 pchg = chg/base*100;
200
201 run;
202 ***** verisian: macro invocation end of <%cfb> *****;
203 *****;
204
205 data adef;
```

Dataset derivations across programs

Select datasets

adam.ad

DatasetReference

adam

adam.adae

adam.ade

adam.adsl

adam.adtte (final)

adtte.sas (260)

Dataset GraphOverview Graph

Datasets: adam.adtte

OriginalResolvedLogDerivation

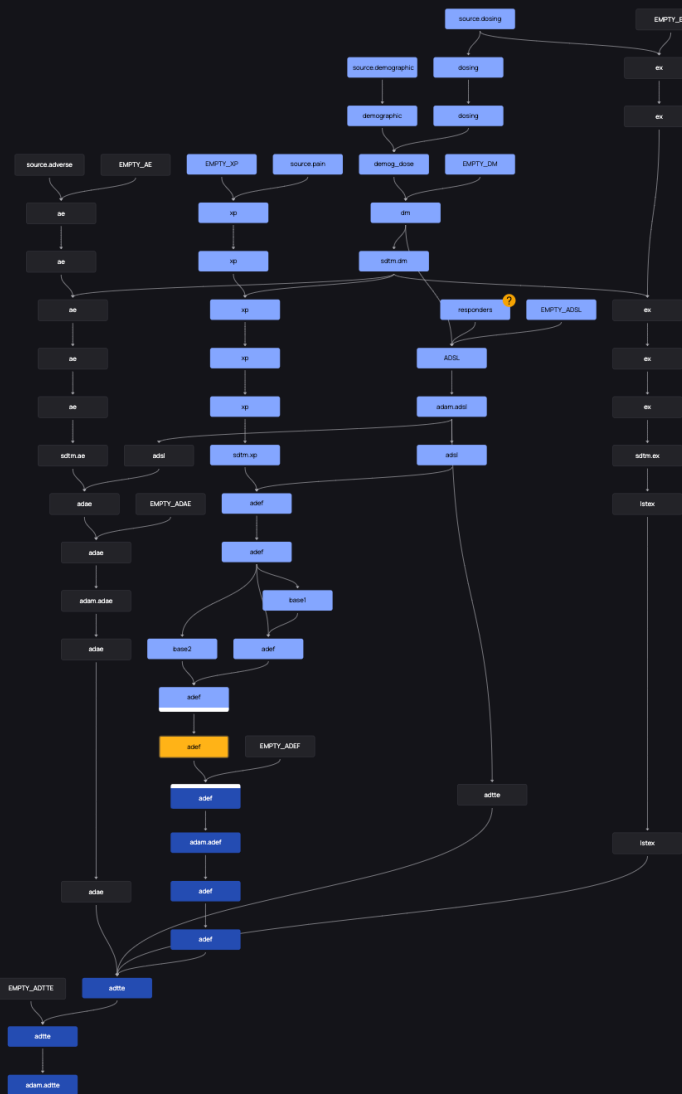
adef.sas (Macro-resolved)

```
106 retain CRIT1 '';
107 attrib ITTFL label="Intent-to-Treat Flag" length=$1.;
108 retain ITTFL '';
109 if 0;
110 run;
111 ***** verisian: macro invocation end of <%make_empty_dataset> *****;
112 *****;
113
114
115 proc sort
116   data = adam.adsl
117   (keep = usubjid siteid country age agegr1 agegrln sex race randdt trt01p trt01pn ittf1)
118   out = adsl;
119   by usubjid;
120
121 data adef;
122   merge sdtm.xp (in = inadef) adsl (in = inadsl);
123   by usubjid ;
124
125   if not(inadsl and inadef) then
126     put 'PROB' 'LEM: Missing subject?-- ' usubjid= inadef= inadsl= ;
127
128   rename trt01p    = trtp
129          trt01pn   = trtpn
130          xptest    = param
131          xptestcd  = paramcd
132          visit     = avisit
133          xporres   = avalc
134   ;
135   if inadsl and inadef;
136   avisitn = input(put(visitnum, avisitn.), best.);
137
138 ***** verisian: macro invocation start of <%dttc2dt> *****;
139 *;
140 *   %dttc2dt(xpdtc, refdt=randdt);
141 *;
142 *****;
143
```

Log Messages

There are no log messages associated with that dataset

Understand derivations across the study

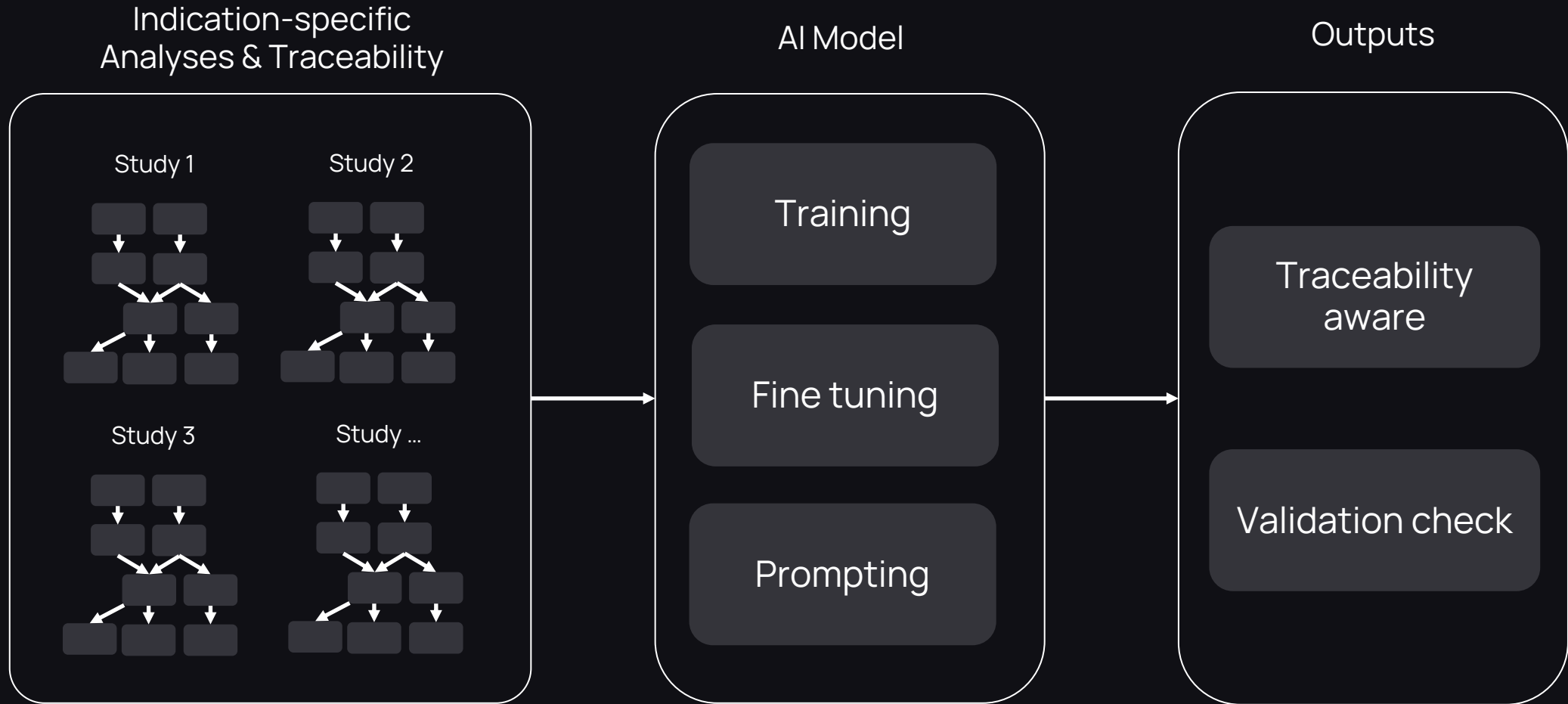


```

Original    Resolved    Log    Derivation
Full Derivation
468      put "WARN" "ING: key variables do not define an unique record." usubjid=;
469
470      retain xpseq;
471      if first.usubjid then
472          xpseq = 1;
473      else
474          xpseq = xpseq + 1;
475
476      label xpseq = "Sequence Number";
477  run;
478
479  * XP.sas;
480  proc sort
481      data=xp(keep = &XPKEEPSTRING)
482      out=sdtm.xp;
483      by &XPSORTSTRING;
484  run;
485
486  * adef.sas;
487  data adef;
488      merge sdtm.xp (in = inadef) adsl (in = inadsl);
489      by usubjid ;
490
491      if not(inadsl and inadef) then
492          put 'PROB' 'LEM: Missing subject?-- ' usubjid= inadef= inadsl= ;
493
494      rename trt01p      = trtp
495             trt01pn    = trtpn
496             xptest     = param
497             xptestcd   = paramcd
498             visit      = avisit
499             xporres    = avalc
500      ;
501      if inadsl and inadef;
502      avisitn = input(put(visitnum, avisitn.), best.);
503
504  *****      verisian: macro invocation start of <%dtc2dt>      *****;
505

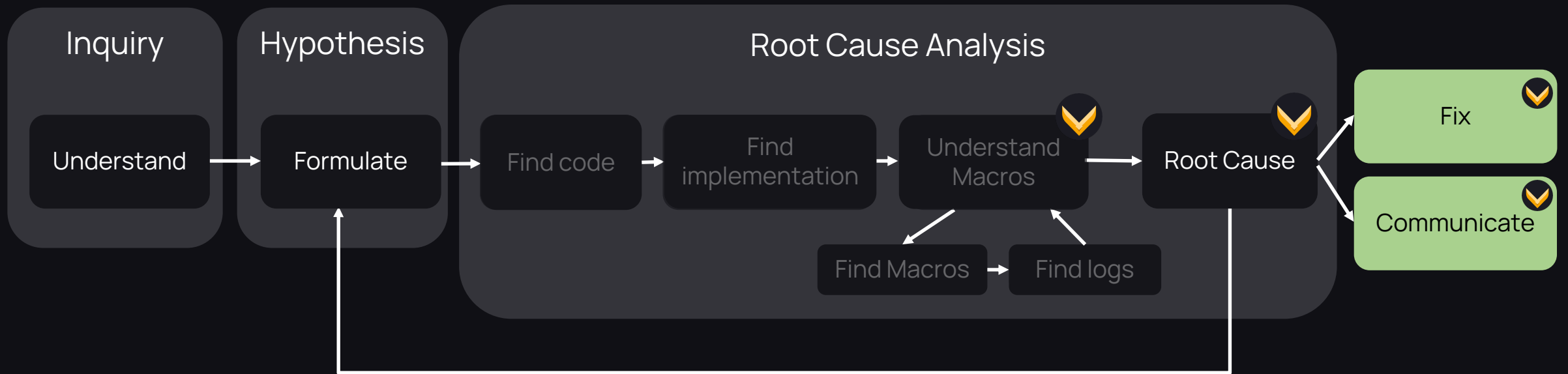
```

Building **custom** AI co-pilots



Inquiry Response Cycle

Verisian AI co-pilots



Explain derivations using Verisian AI co-pilots

Datasets: adam.adtte



```
1219 * adtte.sas;
1220 data adtte;
1221   merge adtte (in = inadtte rename=(randtt=starttdt))
1222         adef (in = inadef)
1223         adae (in = inadae)
1224         lstex (in = inlstex)
1225         ;
1226   by usubjid ;
1227
1228   retain param "Time to first pain relief (days)"
1229         paramcd "TTPNRELFF"
1230         ;
1231   rename trt01p = trtp
1232         trt01pn = trtpn
1233         ;
1234   length srcvar $10. srcdom $4.;
1235
1236   if (.<chg<0) and (adt<astdt or not inadae) then
1237   do;
1238     cnsr = 0;
1239     adt = adt;
1240     evntdesc = put(cnsr, evntdesc.) ;
1241     srcdom = 'ADEF';
1242     srcvar = 'XPDY';
1243     srcseq = xpseq;
1244   end;
1245
1246   else if chg>0 and (adt<astdt or not inadae) then
1247   do;
1248     cnsr = 1;
1249     adt = adt;
1250     evntdesc = put(cnsr, evntdesc.) ;
1251     srcdom = 'XP';
1252     srcvar = 'XPDY';
1253     srcseq = xpseq;
1254   end;
1255
1256   else if (.<astdt<adt) then
1257   do;
1258     cnsr = 2;
1259     adt = astdt;
1260     evntdesc = put(cnsr, evntdesc.) ;
1261     srcdom = 'ADAE';
1262     srcvar = 'ASTDY';
1263     srcseq = aeseq;
1264   end;
1265
1266   else
1267   do;
1268     cnsr = 3;
1269     adt = trtedt;
1270     evntdesc = put(cnsr, evntdesc.) ;
1271     srcdom = 'ADSL';
1272     srcvar = 'TRTEDT';
1273     srcseq = .;
1274   end;
1275
1276   aval = adt - startdt + 1;
1277   format startdt adt yymmdd10.;
1278 run;
```



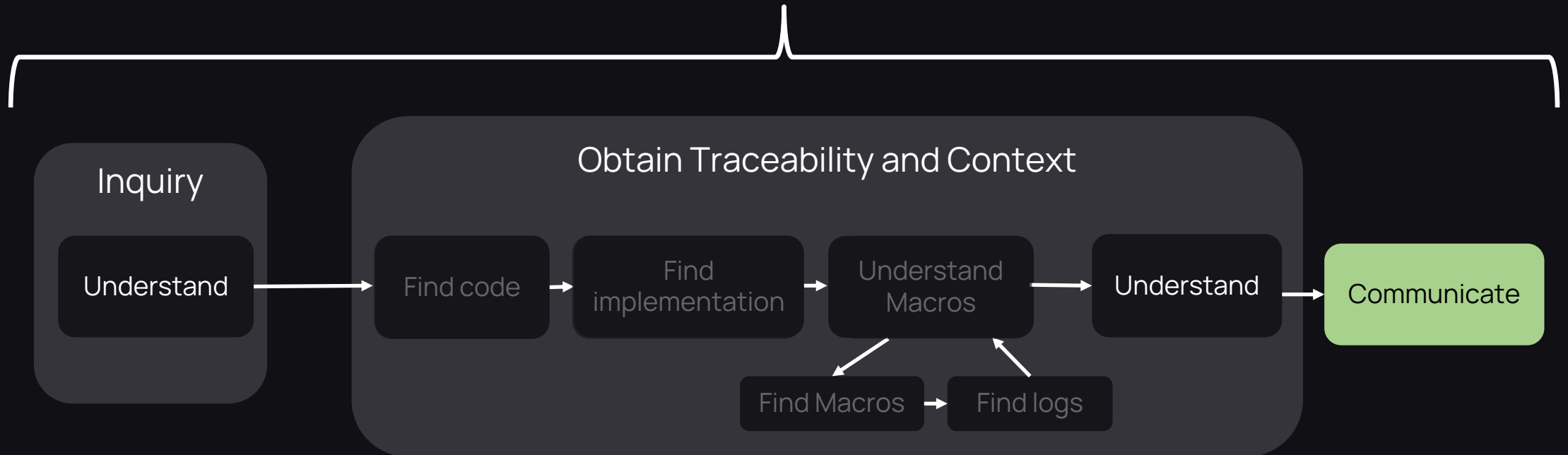
1. The code performs a merge operation to combine data from various sources (ADDTE, ADEF, ADAE, LSTEX) based on the USUBJID variable.

2. The code assigns and retains parameter values for the "Time to first pain relief (days)" with a standard code "TTPNRELFF".

3. The code classifies events (chg > 0 or < chg < 0) based on therapy start (trtp, trtpn, srcvar, srcdom), assessment start date (astdt), assessment end date (adt), and therapy end date (trtedt).

Explain and Communicate

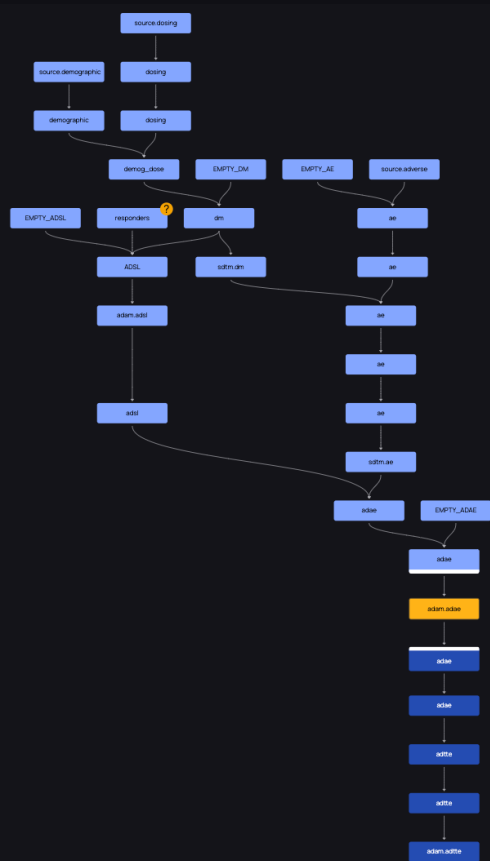
Verisian AI co-pilots



Answer Inquiries with Verisian AI co-pilots

Inquiry:

How is the treatment-emergent time frame defined? Provide relevant code snippets.



```
Original Resolved Log Derivation
Full Derivation

627 *****
628
629 if length(trim(aeendtc))=10 and index(aeendtc,'-')==0 then aeendtc = input(aeendtc, yymmdd10.);
630 else if length(aeendtc)=16 and index(aeendtc,'--')==0 and index(aeendtc,'-')==0 then do;
631   aeendtc = input(trim(aeendtc)||"00", e8601dt19.);
632   aeendtc = datepart(aeendtc);
633   * optionally add formats;
634   format aeendtc datetime16.;
635 end;
636 if .<aeendtc>trtsdt then aendy = aeendtc - trtsdt;
637 else if aeendtc=trtsdt then aendy = aeendtc - trtsdt + 1;
638 * optionally add formats;
639 format aeendtc yymmdd10.;
640 *****          verisian: macro invocation end of <end2dt>          *****;
641 *****
642
643
644 if index(AEDECOD, 'PAIN')>0 or AEDECOD='HEADACHE' then
645   CQ01NAM = 'PAIN EVENT';
646 else
647   CQ01NAM = '          ';
648
649 aere1n = input(put(aere1, $aere1n.), best.);
650 aesev1n = input(put(aesev, $aesev1n.), best.);
651 relgr1n = (aere1n=0); ** group related events (AERE1N=0);
652 relgr1 = put(relgr1n, relgr1n.);
653
654 * Event is considered treatment emergent if it started on or after ;
655 * the treatment start date. Assume treatment emergent if the start;
656 * date is missing (and the end date is either also missing or on or;
657 * after the treatment start date) ;
658 trtemf1 = put((astdt=trtsdt or (astdt<=.z and not(.z<aeendtc-trtsdt))), _only.);
659 if astdt=trtsdt then
660   trtemf1 = 'Y';
661 format astdt aeendtc yymmdd10.;
662 run;
663
664 * adae.sas;
665 data adae;
666   retain $adaeKEEPSTRING;
667   set EMPTY_adae adae;
668 run;
669
670 * adae.sas;
671 proc sort
672   data=adae(keep = $adaeKEEPSTRING)
673   out=adam.adae;
674   by $adaeSORTSTRING;
675 run;
```



```

        trtemfl = put((astdt>=trtsdt or
(astdt<=.z and not(.z<aendt<trtsdt))),
_0nly.);
        if astdt>=trtsdt then
            trtemfl = 'Y';

```

This code sets the treatment-emergent flag (`trtemfl`) to 'Y' if the start date of the adverse event (`astdt`) is greater than or equal to the treatment start date (`trtsdt`). If the treatment start date is missing, the adverse event is considered treatment-emergent if the end date is either missing or is on or after the treatment start date.

Programmers

Faster developing, debugging, and inquiry response

Sponsors

Faster iterations, greater submission quality and confidence,
faster time-to-market

Regulators

Faster and easier review

Inquiry Response one example of Traceability in Action

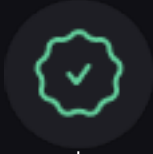
But there are many other processes in which
Traceability paired with AI
can introduce a paradigm shift

Based on Traceability, Empowered by AI



Builder

Build studies faster and at higher quality



Validator

Ensure optimal quality with automated quality control



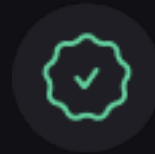
Submitter

Submit transparently with quality and confidence

Clinical Trial Analyses using Verisian



Faster and better build



Automated Quality Control



Submit in confidence with full transparency

Raw

SDTM

ADaM

TLFs

Submission

Review