Paper DH01

Don’t Let Your Data Handle You: A Novel Approach to Clinical Programming

Jorine Putter, Grünenthal GmbH, Aachen, Germany
Michael S Rimler, GlaxoSmithKline, Cincinnati, Ohio, US

We propose a data handling technique which we argue improves the efficiency of both data query tracking and statistical programming tasks. The method proposed relies on programming teams adopting the philosophy of ‘programming to what is expected, not what is observed’ in the data. When this technique is implemented, any unexpected data in the ‘raw to SDTM’ mapping is filtered out prior to SDTM database mapping and exported into an external file for further review.

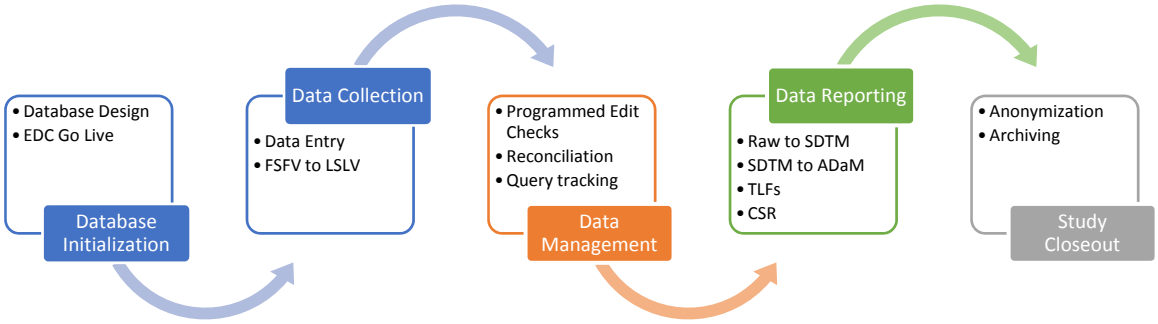
The external file created serves three purposes. First, all unexpected data is conveniently packaged for Data Management (DM) review after each full execution of downstream programming. Second, if any unexpected data persists due to the inability for data monitoring to sufficiently clean the data (e.g., closed or unresponsive sites), the external file can be leveraged to modify the unexpected data in lieu of programmatic hardcoding. By appending the external file with added fields for updated values, one can prescribe how a problematic data field should be handled. When imported by SDTM programming, agreed upon (and documented) changes to the collected data can be performed without explicitly hardcoding within downstream programming. Third, the file serves as an historical log for data query tracking. If an issue persists, it is not reissued to DM for query, as it will be tracked that the issue was already communicated to DM for review.

We propose that this data filtering is processed with a new set of programming on the raw database. An additional benefit of this approach is that downstream SDTM, ADaM, and TLF programming should execute properly without further modification under this solution. If additional issues impact the execution of, or interpretation of results from, downstream programming, it is the raw database processing code that would be updated, not the downstream programming. For reporting milestones during an ongoing study (e.g., dry runs), we argue that this solution will (1) require much less time to be invested in ‘programming around’ data issues, (2) improve the interpretability of results, and (3) provide a neatly packaged set of data issues and resolutions which can accompany the dry run package.

SECTION 1: BACKGROUND

During the life of a clinical trial database, there are three primary processes that operate on the data: Data Collection, Data Management, and Data Reporting. Figure 1 presents a linear representation of these processes as well as Study Closeout activities after Data Reporting is completed.

Figure 1. Illustration of the End-to-End Process of Clinical Trial Data



During Data Collection, sites input patient data into the electronic case report form (eCRF) such as demographics, vital signs, ECG results, lab test results, and/or questionnaire data. As with any data entry process, newly collected data is at risk for entry error. Although an electronic data capture (EDC) system may have automatic entry checks

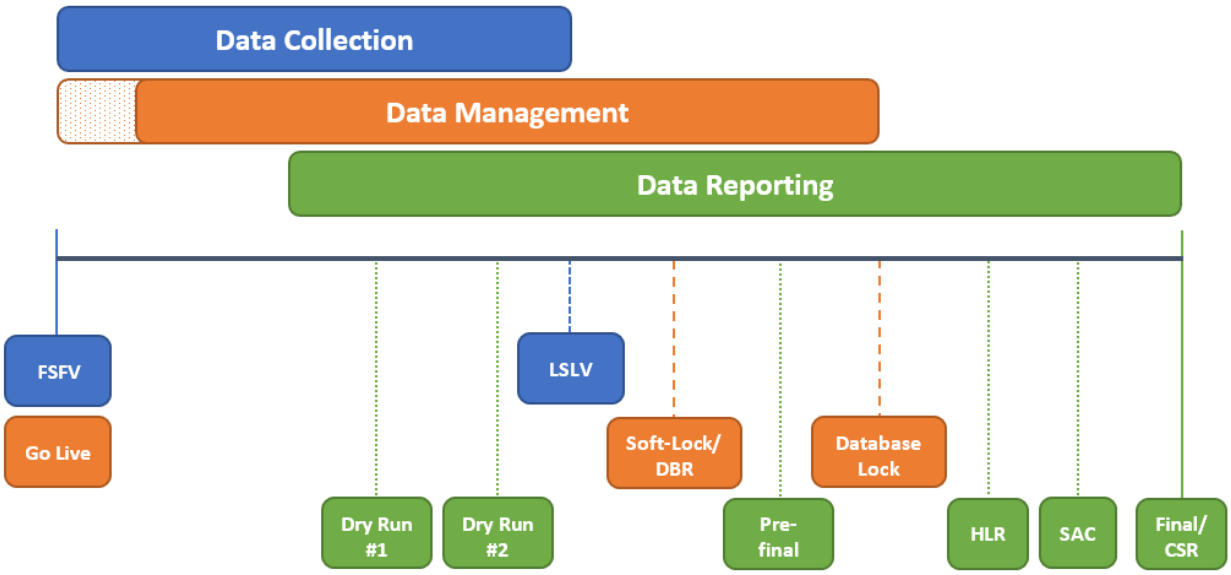
built in (for example, not allowing birthdates in the future), entry errors cannot be fully mitigated through EDC designed data entry safeguards. This leads us to the next process of Data Management.

Data Management involves reviewing the collected data to ensure its completeness and quality. Data managers review data as it is collected and issue queries to sites for any findings. This review may include automatic query generation embedded in the EDC system which allow the data to be entered but generates an automatic query for review and resolution. Data Management may also employ programmed edit checks such as adverse event (AE) start dates occurring prior to AE end dates or flagging vital signs that are seemingly out of range. Records flagged through programmed edit checks may trigger DM to query the site and result in either modification of the data or confirmation that the data is correct as entered. A third component of Data Management review typically involves reconciliation of externally provided results such as lab panels or central ECG readings. A fourth and, for our purposes, final component of Data Management review involves reviewing potential data issues flagged by Statistics and Programming (S&P) groups during the development of dataset and display programming. Data managers may receive reports of potential data issues from S&P and must determine if a data query should be raised to the site for each finding.

The third process, Data Reporting, is typically performed by S&P groups. At the highest level, S&P is tasked with ingesting the data as collected, processing it, analyzing it, and reporting the results in a readable format to support the objectives of the study protocol in adherence with the statistical analysis plan (SAP). In a trial that complies to CDISC standards, S&P would first map the data from its collected (raw) form into an SDTM compliant database. Although the SDTM mapping process may instead be conducted by Data Management, the assumption of S&P performing the SDTM mapping does not impact our analysis in a significant manner. S&P would then transform the data from the SDTM design into an ADaM compliant database design to ensure the database is 'analysis-ready'. Finally, S&P uses the ADaM database to generate the analysis results via tables, listings, and figures (TLFs) specified in the SAP.

In practice, however, these processes run concurrently and are not simply linearly related as we have illustrated in Figure 1. Data Management does not wait for all data to be collected before commencing with their cleaning and reconciliation tasks. S&P does not wait for a clean database prior to developing the programming infrastructure needed to map the data and produce the TLFs. Therefore, Figure 1 is not an adequate representation of how the clinical trial database matures because these three processes are not serially related. A more appropriate representation of the database lifecycle is illustrated in Figure 2, the concurrent nature of which creates both additional challenges for the groups and additional opportunities for efficiency and quality improvements.

Figure 2. Updated Illustration of the End-to-End Process of Clinical Trial Data



In this representation, Data Collection remains active from the first patient's first visit until the last patient's last visit. Data Management activities commence shortly after the first data is collected, are ongoing while data continues to be

collected, and continue after the last patient's last visit until database lock (DBL) is declared. Data Reporting activities begin very early in the study lifecycle and continue through the completion of the TLFs supporting the clinical study report (CSR). In Figure 2, S&P is responsible for several reporting milestones, the first of which is Dry Run #1, reported while the trial is ongoing. This Dry Run reporting is performed prior to the database being declared clean and may include a fully validated package SDTM, ADaM, and TLFs.

In support of instream data reporting (dry runs, interim analyses, futility analyses, etc.), S&P groups often face the challenge of preprogramming and reporting fully validated datasets and TLFs on unclean data. Prior to declaration of a clean database, any data extract is likely to contain data issues. Although these data issues may already be known to DM and queries issued, S&P programming must report on the unclean data extract and may observe either execution errors or anomalous interpretations of results. In the case of execution errors, programs must be updated (i.e., 'program around the data issue'). In the case of anomalous results (e.g., a patient with a recorded weight of 220kg vs 100kg (220lbs)), the study team must be consulted on how to handle the data for the reporting milestone and then programming must be updated to account for the decision. A significant amount of resource may be required to identify the source of issues, plan a resolution, and ultimately resolve the issues appearing in the extracted database.

We propose a solution which mitigates these challenges on downstream programming maintenance, while also improving the Data Management review process in terms of quality and effort by DM resources. Prior to outlining the solution, we present a few examples of data issues that may be observed in databases prior to declaration of a clean database and DBL.

SECTION 2: EXAMPLES OF UNEXPECTED DATA ISSUES

In this section, we present a few examples for illustration of the types of data issues that S&P teams encounter when preparing a data cut for a pre-DBL reporting milestone such as a dry run. Although some of these may be prevented with an EDC entry check disallowing certain values from being entered, it is reasonable to assume that not all systems will have a specific check in place. Hence, these issues may appear in the database that S&P teams process for a reporting milestone, even if an automatic query is generated by the EDC system.

Example 1. The concomitant medications data in Table 1 contains two data issues. For Patient 1001, the second medication (Medication 2) has a medication stop date prior to the medication start date. Perhaps a data query has already been issued for this type of data issue, either by an EDC automatic query or a DM programmed edit check. However, the data was still transferred as entered and S&P teams must determine how to handle this data for analyses such as calculation of duration of medication or average daily doses.

Table 1. Example Concomitant Medications Data

Patient Number	Medication	Medication Start Date	Medication Stop Date
1001	Medication 1	1-Jun-18	UNK
1001	Medication 2	14-Jun-18	13-Jun-18
1002	Medication 1	5-Jun-18	8-Jun-18
1002	Medication 2	7-Jun-18	xx-Jun-18
1003	Medication 2	18-Jun-18	21-Jun-18
1004	Medication 1	3-Jun-18	30-Jun-18
1004	Medciation 1	3-Jun-18	30-Jun-18
1004	Medication 2	10-Jun-18	14-Jun-18
1005	Medication 2	8-Jun-18	Ongoing
1005	Medication 1	4-Jun-18	4-Jun-18

Example 2. A second example, also in Table 1, is the seemingly duplicate entry of Patient 1004's concomitant medication data. Based on the information provided, it appears that the medication was entered in duplicate with a slightly different verbatim text (Medication) in the second record. Duplicate records can be problematic in downstream dataset mapping (e.g., merge issues) and TLF reporting (e.g., frequency counts or total days exposed). Again, even if DM has a programming system in place to identify the incidence of duplicate entries, but the issues may still be present in the database transferred to S&P teams.

Example 3. Our third example is less likely to be identified with typical data management cleaning processes other than manual review of critical data points, as the data issue is an inconsistency across multiple data modules.

Typically, these types of issues are flagged either during manual review of the data or downstream when S&P teams begin to repackage the data for analysis purposes.

In Table 2, Patient 2003 has 3 recorded adverse events (AEs): dizziness, stroke, and death. Dizziness began on July 10 and ended on July 20. The patient experienced a stroke on July 20 (duration of 1 day) and death which started on July 18 and ended on July 20. Within this module alone, one may question the death AE record where the event of 'death' had a duration of 3 days. However, when reviewing the patient's disposition data on the end-of-study page (see Table 3), the patient is recorded to have died on July 18 and discontinued on July 19. If this death date is correct, then we have a patient experiencing a stroke after discontinuing from the study and after experiencing death. The AE stop dates of all 3 AEs, AE start date for death, and two dates in the disposition data should all be queried. Until resolved, however, S&P teams must determine how to properly handle this data anomaly and modify programming accordingly. Discontinuation and death dates are often used in algorithms for determining the end of treatment phase or imputing partial dates. In addition, dataset programming may fail to validate due to the independent programmer pulling death date from AE end date instead of the end-of-study domain.

Table 2. Example Adverse Event Data

Patient Number	AE Term	AE Start Date	AE Stop Date
2002	Fatigue	1-Jul-18	8-Jul-18
2003	Dizziness	10-Jul-18	20-Jul-18
2003	Stroke	20-Jul-18	20-Jul-18
2003	Death	18-Jul-18	20-Jul-18
2004	Nausea	1-Jul-18	5-Jul-18
2004	Fatigue	1-Jul-18	5-Jul-18

Table 3. Example End of Study (Disposition) Data

Patient Number	Discontinuation Date	Death Date
2001	-	-
2002	-	-
2003	19-Jul-18	18-Jul-18
2004	-	-
2005	-	-

Example 4. Consider the exposure and lab data for Patient 3004 in Table 4. The first dose date/time information for each patient is presented, along with lab draws performed at the same visit. In this example, the lab draw is intended to be the baseline draw, but Patient 3004 has a recorded lab time of 9:03, relative to a dose time of 9:02. Therefore, according to the recorded data, this patient was dosed prior to the lab draw. Depending on the definition of baseline outlined by the protocol or analysis plan, this record may be ineligible for baseline and the patient may not have a baseline value for any lab parameters analyzed by this draw. Note that Patient 3006 is also problematic because we do not know if they were dosed prior to or after the lab draw. The recorded time for first dose and lab draw was within the same minute (9:01). Data for both Patient 3004 and 3006 should be queried. S&P must consult with the study team to determine how to handle these data and update programming accordingly.

Table 4. Example Exposure and Lab Data

Exposure Module

Patient Number	EX Start Date	Ex Start Time
3001	1-Jul-18	9:00
3002	10-Jul-18	9:15
3003	20-Jul-18	8:55
3004	18-Jul-18	9:02
3005	1-Jul-18	8:47
3006	1-Jul-18	9:01

Lab Module

Patient Number	Lab Start Date	Lab Start Time
3001	1-Jul-18	8:58
3002	10-Jul-18	9:12
3003	20-Jul-18	8:50
3004	18-Jul-18	9:03
3005	1-Jul-18	8:30
3006	1-Jul-18	9:01

Example 5. Our final examples involve the vital sign data presented in Table 5. First, Patient 4002's height and weight are disproportionate to a typical human being. The calculated body mass index with these measurements is 43.4kg/m^2. In addition, the height and weight of Patient 4003 appears to be captured in incorrect units, perhaps feet and pounds, respectively. Finally, the systolic blood pressure for Patient 4004 is lower than the diastolic measurement. For each of these potential issues, DM may have checks to query sites on the validity of the data, but S&P teams will very likely observe these anomalies in the extracts provided for instream reporting. The study team will need to determine how to handle the data and programming (SDTM, ADaM, and/or TLF) will need to be updated accordingly.

Table 5. Example Vital Sign Data

Patient Number	Systolic Blood Pressure	Diastolic Blood Pressure	Height (cm)	Weight (kg)
4001	128	80	165	75
4002	125	78	140	85
4003	130	82	6	220
4004	80	120	185	92
4005	120	90	172	83

These are a few examples of data issues that S&P teams may encounter prior to a clean database and achieving DBL, but they are not exhaustive of the types of issues that can impact programming execution and/or interpretation of results. Perhaps a patient has a recorded concomitant medication coded as an oral corticosteroid, but the site has entered the route of administration as injection. Or, perhaps the parameters returned by a central laboratory are either coded unexpectedly or simply contain different parameters (or units of measure) than planned. For each of these examples, it is reasonable to foresee that the data issues would exist in the data extract and be included in the input data for S&P SDTM, ADaM, and TLF programming. These data issues could cause execution errors within the downstream programming. However, even if programs execute without error, the interpretation of results could be impacted due to the potential data issues. In either case, S&P teams may need to update their programming infrastructure to account for the data issues and implement any interim solutions decided by the study team.

In the next section, we propose a solution whereby more effort is placed in the upstream programming to reduce the likelihood that downstream programming is critically impeded by processing unclean data. The proposed solution relies on a simple concept which is to 'program to what is expected, not what is observed'.

SECTION 3: A PROPOSED SOLUTION

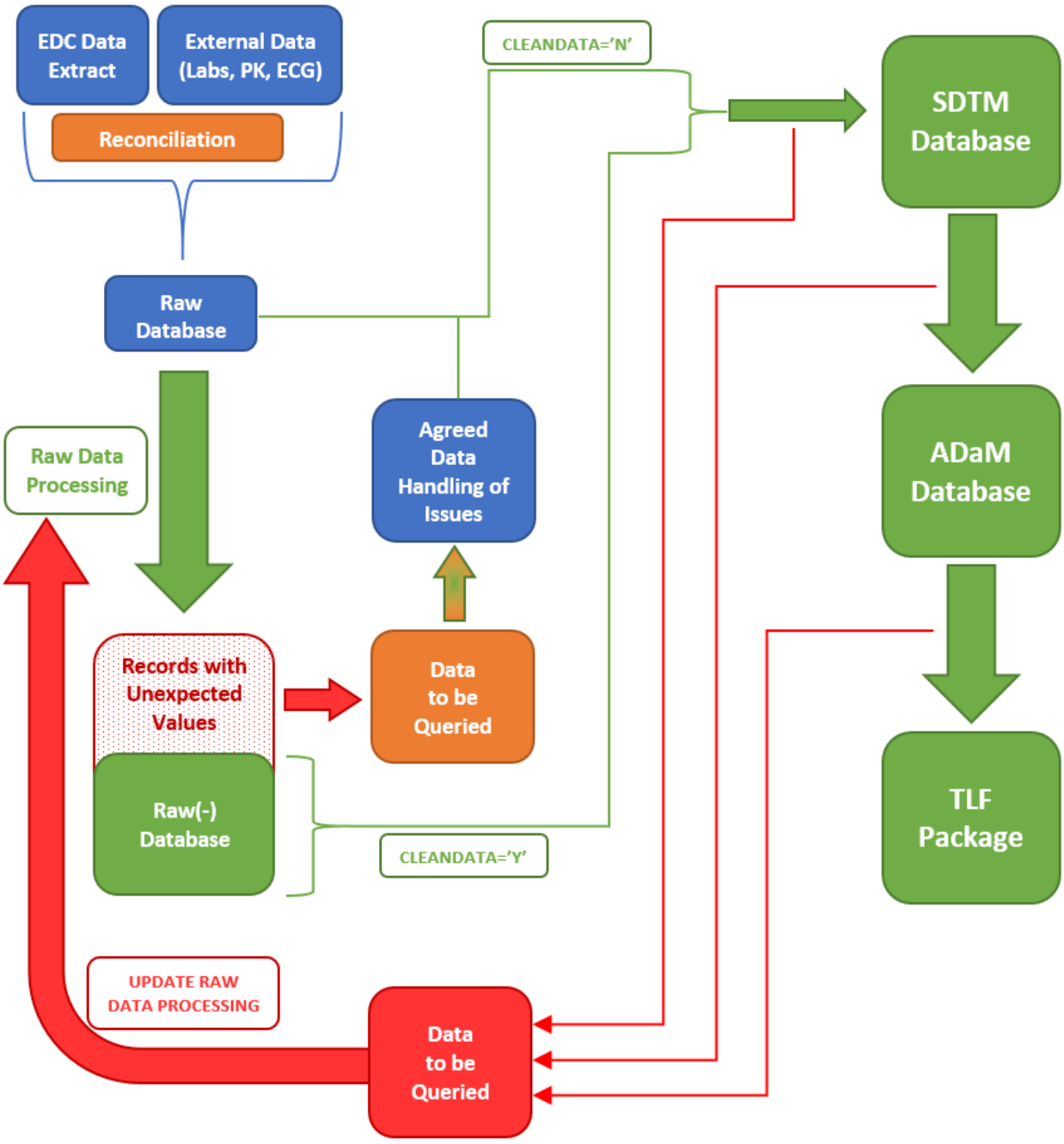
We propose a solution to the challenges outlined, namely that S&P groups are often required to deliver fully validated reporting packages (datasets and TLFs) not just prior to databases being declared clean, but even prior to the end of the data collection period. The programs which are developed to create or validate each dataset and display are typically preprogrammed on one set of data. Near the delivery milestone, e.g. a dry run, Data Management will extract and transfer the raw database to S&P, often excluding any data collected after a predetermined data cutoff (DCO) date. The set of data in this transfer will contain both new and modified records, relative to the database that S&P preprogrammed on. However, S&P is expected to execute every program without error and fully validate each output (dataset and TLFs). It is possible that the new or modified data generates an error, either in execution or in analysis results interpretation, which requires investigation and possible remediation by S&P to successfully deliver.

Our solution relies on two critical features. First, an innovation step of S&P managed programming which processes the raw data and identifies potential data issues impacting downstream programming or interpretation. Second, a feedback mechanism for downstream programming to inform this initial raw data processing of additional potential data issues that are difficult to predict at the outset of the study. These types of data issues may remain hidden until analysis datasets or TLFs are generated and reviewed. For example, a lab draw recorded 1 minute after first dose when it is expected to be performed prior to first dose. In the analysis, it may be observed that this patient had no baseline values and further investigation would identify the reason. Our solution requires that this information be fed back to the initial process which process the raw data prior to SDTM mapping. The overall process, including these two critical components, is depicted in Figure 3.

At first glance, the process appears quite complicated. However, note that all complexity lives between the delivery of the raw database from DM to S&P and the SDTM mapping process. Typically, the group that maps the SDTM database inputs the raw database as transferred from Data Management, but this solution implements an

intermediate programming step prior to SDTM mapping which is the critical innovation of this solution. Let's walk through this process one step at a time.

Figure 3. Proposed Data Handling Process



Section 3.1: Raw Data Processing

Following the transfer of the raw database from DM to S&P, a newly proposed set of programming identifies potential data issues based on planned edit checks. There may be overlap between this programming and automatic or programmed edit checks performed by DM, but we contend that data issues may still infiltrate the transferred database even if a query has already been issued. Any raw record flagged by this programming is separated from the

raw database. The output of the programming contains two mutually exclusive and exhaustive sets of data. First, a "raw(-)" database which is clean of any data issues, as defined by the S&P programmed edit checks. The remaining records, i.e. records with unexpected values, are contained in the second set of data for potential query.

The next step in the process sends the records with unexpected values back to DM for potential issuance of data queries. We suggest that the mechanism for tracking these data issues is via an external file with the example design presented in Table 6. The actual design may be situation specific, so we merely describe characteristics we feel are important to maximize the functionality of the external file.

Table 6. Example of External File for Tracking Raw Data Issues (VITALS DOMAIN)

PATID	SYSBP	DIABP	HGT	WGT	RawVar	UpdVal	Comment	PRVFLAG	RSLVD	QRY
4002	125	78	140	85	HGT	170		Y	N	N
4002	125	78	140	85	WGT			Y	N	N
4003	130	82	6	220	HGT	183		Y	N	Y
4003	130	82	6	220	WGT	100		Y	N	Y
4004	80	120	185	92	SYSBP	120		N		
4004	80	120	185	92	DIABP	80		N		

First, we suggest a single spreadsheet file such as a multiple sheet Microsoft Excel ® file. Each raw domain would have its own sheet, e.g. DEMO, AE, or CONMED. Within each sheet, there would be one record for each unexpected data field flagged by the raw data processing. In the first 5 columns in Table 6, we see the complete data, as collected, for problematic records. For Patient 4002, since both height and weight are flagged, two records are created.

Second, in addition to outputting all collected fields associated with the problematic raw record, 3 fields are added. The first field (RawVar) is populated programmatically with the variable name of the raw data field which triggered the extract (e.g.,'SYSBP'). The second and third fields would be created programmatically but populated manually after review of the output. One field contains the updated value (UpdVal) as determined by the study team and the other field (Comment) contains an explanation for the updated value.

Third, we propose three additional fields to track the data query process. The external file would be a running log, updated and appended at each data cut. If the S&P raw data programming flag had previously flagged this same field, then PRVFLAG='Y', else if a new issue then PRVFLAG='N'. Records that were previously flagged and continue to be flagged are considered unresolved (RSLVD='N'). If a previously flagged record no longer gets identified by raw data programming, then RSLVD='Y'. Finally, if this file is provided to Data Management, they can identify which records have been queried and which will not be queried. Non-queried data and unresolved data (regardless of the reason for non-resolution) will require remediation at the analysis level by the study team. With this design, the external file not only tracks data issues flagged by S&P, but also provides an historical record of the data cleaning process. Furthermore, it provides awareness to S&P teams as DBL approaches to prepare for handling anomalous data that may persist post-DBL.

As the study approaches DBL, the database becomes cleaner and the number of records filtered out should approach zero. However, there may exist persistent data anomalies (RSLVD='N') which require review and a resolution in place on how to handle in downstream programming (SDTM, ADaM, and/or TLF).

Section 3.2: SDTM Mapping

At this stage in the process, S&P has access to three databases: the originally transferred raw database, an external file containing all problematic raw data records (with proposed updated values as agreed by the study team), and a raw(-) database containing only S&P declared clean records. As depicted in Figure 3, SDTM programming can be developed to either input the clean raw(-) database or the full database, the latter of which would allow for merging in proposed data changes for problematic records.

In the first case (CLEANDATA='Y'), it is expected that SDTM programming should execute without error. However, the resulting SDTM databases will exclude all problematic records. Hence, review of the data and analysis results must take this into consideration, but the likelihood of error-free execution with existing SDTM code is higher on an updated raw data extract.

In the latter case (CLEANDATA='N'), all data is included, but identified problematic data fields are modified as prescribed in the external data file. One question that arises how to properly map these changes into the SDTM database design. We suggest that simply changing the SDTM data and excluding the collected value could be seen by regulatory bodies as less than transparent, even if the changes are well documented. On the other hand, we do suggest that the data change occur at the SDTM level so ADaM programming remains independent of the process. So, how do we get the data changes into SDTM without excluding the original collected data? Using the example of Patient 4004 in Table 6, we propose that the database of anomalous data, when merged in, would set VS.VSORRES=120 for systolic blood pressure and a SUPPVS record would be created with QVAL='80' and an appropriate QNAM and QLABEL to uniquely identify it as the originally created CRF value. For example, QNAM='CRFSYSBP' and QLABEL='Original eCRF collected SYSBP'. Diastolic blood pressure can be handled in a similar fashion for this patient. Of course, values of QNAM and QLABEL are specific to the study, CRF design, CRF domain, and CRF field. We also propose that any record included in the external file that is unresolved, whether values are changed or not, should have a supplemental record created that identifies it as a problematic record/field within the SDTM database for traceability.

Finally, Figure 3 depicts a feedback mechanism (red line) back to the raw data processing. If the SDTM programming group encounters any problematic data unforeseen by the current raw data processing, we propose that the appropriate action is to update the raw data processing code rather than the SDTM code. The newly identified problematic record/field would be added to the external file and sent to Data Management for query, rather than updating SDTM programming to 'program around' the data issue.

Note that resolution of the data issue is not required for the reporting milestone to be achieved. Rather, an update to the raw data processing code and subsequent rerun will include this record in the external file. Therefore, if this record is desired (or required) to be included in the data reporting package, the study team can determine the proper handling of the data field, amend the external file, and execute the SDTM mapping again with CLEANDATA='N'. This second round of SDTM mapping programming should run without issue and the data issue continue to be tracked via the external file.

Section 3.3: ADaM Mapping and TLF Generation

The next steps in the process depicted in Figure 3 are ADaM database mapping and TLF generation. In theory, all data issues are either excluded from the SDTM database (CLEANDATA='Y') or handled as agreed by the study team (CLEANDATA='N'). In either case, ADaM and TLF programming is expected to execute without error. However, this is not always the case as many data issues are not uncovered until data is prepared for analysis, particularly inconsistencies across domains.

Similar to SDTM programming, if a particular data field causes execution error in ADaM or TLF programming, it should be fed back for updating of S&P raw data processing, rather than updating the ADaM or TLF programming directly. Even if programming executes without error but TLFs reveal that the minimum RR interval of patients in the study is 0.98, raw data processing should be updated to query if records with such low RR are collected as 0.98msec or 0.98sec.

In addition, we propose to include two analysis flags in all ADaM datasets to facilitate flexibility in reporting, the naming of which is admittedly arbitrary. First, ANL98FL can be added to flag any record that was excluded from the raw(-) database. If all problematic records excluded from the raw(-) database have an SDTM supplemental record created (as proposed), then this analysis flag can be derived directly from the SDTM database and is traceable. Second, ANL99FL can be added to flag any record that has the collected value changed via the external value (UpdVal). Again, if the SDTM supplemental domains are utilized as proposed, then ADaM programming can generate this analysis flag with the SDTM database alone.

If ANL98FL and ANL99FL are included, then TLF programming can leverage this information to run analyses with either all data, all data that was not changed from originally collected data (ANL99FL ne 'Y'), or all data not flagged as problematic and unresolved (ANL98FL ne 'Y'). Care would have to be taken for derived records where a component contributing to the derivation is problematic, but that is a decision for the study team to make when deriving ANL98FL or ANL99FL for derived records.

Section 3.4: Additional Role of External File

The external file that is created and maintained by S&P raw data processing serves multiple roles. First, it is a running log of all S&P identified data issues with a history of what was identified by DM as queried, which issues were

resolved, and which issues remain unresolved. Second, it allows for transparent modification of the collected data without hardcoding within SDTM, ADaM, or TLF programming directly. Third, it can accompany a submission package (e.g., appended to the Clinical Study Data Reviewer's Guide (cSDRG)) to document and explain changes to the collected data required for proper reporting.

SECTION 4: BENEFITS AND CHALLENGES OF THIS APPROACH

We are not proposing that this solution is a one-size-fits-all methodology that resolves all issues facing S&P and Data Management groups when handling data in an ongoing trial. However, we contend that the solution exhibits several benefits which have the potential to improve the efficiency of the data handling process. Although we also identify a few challenges that must be considered, we argue that if these are offset by the benefits, then DM and S&P departments should coordinate their efforts to implement a solution in the spirit of what we have discussed.

Section 4.1: Cultural Shift

Perhaps the greatest challenge to this solution is the need for a cultural shift within the organization. It is widely agreed and acknowledged that S&P teams are not accountable to find data issues. However, in practice data issues are the biggest pain point for S&P teams to achieve delivery milestones, particularly prior to DBL. Not only must S&P develop mechanisms to ingest, analyze, and interpret these problematic data issues, but they also need to ensure that there is a transparent lineage of each raw collected value to an analysis result. Put another way, S&P must be able to prove that an anomalous analysis result is due to dirty data and not a programming issue. Similarly, a big pain point for DM involves spending double efforts to manage the data issues flagged by S&P groups and reconciling them with their typical query tracking processes. Therefore, by implementing the proposed solution, a mindset shift is required. S&P teams must invest time in upstream programming efforts (raw data processing) to facilitate smoother downstream programming activities as well as provide a seamless data issue management mechanism for their DM colleagues when they are reporting potential problematic data issues.

Furthermore, as with any operational change, one must carefully consider clear task ownership assignment between potential different S&P groups and DM groups, particularly if they are housed across multiple organizations (e.g., sponsor and multiple contract research organizations (CROs). Embedded in this is also the necessity of streamlined communication channels to resolve critical roadblocks especially during tight timelines. For example, when a new data extract is provided to support a delivery milestone, any new data issues identified by S&P programming will require an agreed remedy before the deliverable package can be finalized.

We argue that these challenges are worth facing given the potential benefits outlined in the remainder of this section.

Section 4.2: Data Query Tracking

In this solution, S&P provides a conveniently packaged set of all unexpected data issues to DM. This unexpected data can be annotated by DM to ensure that once a query has been raised, the specific query will not be flagged as a new query for the remainder of the trial. This means that the issues provided by S&P need to be cross-checked against DM's own identified issues once and only once. In addition, since the issues are the result of programmed edit checks on the raw data, this data query log will not contain duplicate entries generated by disparate programmers encountering the same data issues independently and re-entering.

This approach will also ease with the management unexpected data over the study lifecycle. Previously identified issues will remain in the running log and flagged as resolved/unresolved with respect to the S&P programmed edit checks. By tracking the issues over the lifecycle, study teams can prepare for data issues which are expected to persist and remain unresolved through to DBL and decide the appropriate data handling rule(s).

Finally, this running log of data issues flagged by raw data processing can be leveraged to calculate diagnostic metrics on the data query process. For example, one can report out the number of open queries, the resolution rate of queries, or the incidence of clean data modules from the S&P perspective. This information can provide valuable information to DM on their data cleaning processes as well as identify if the study is on a critical path as DBL approaches.

Section 4.3: Efficiency gains with SDTM, ADaM, and TLF programming

Once the unexpected data issues have been separated from the "clean raw data", SDTM programs can be created to expected data, which saves time as there shouldn't be a need to program around any unexpected data issues. When SDTM datasets are required on all data, problematic data issues can be handled with the external file as described in

Section 3.2. This minimizes the need to modify any SDTM programs, apart from just running it with the correct input parameters, which is perhaps the greatest benefit of this solution – minimizing the need to maintain downstream code after an updated raw data transfer from DM to S&P.

Similarly, when anomalous data is encountered through ADaM and TLF programming, the data in question is added to the external file and the data can be handled through the external file instead of having to amend ADaM or TLF programs. Therefore, the initial investment in the upstream programming for raw data processing will significantly alleviate program maintenance throughout the lifecycle of the trial. We argue that this will result in significant cost savings for S&P functions as they will not expend unnecessary resources resolving unclean data driven issues in the downstream programming to meet a delivery milestone.

Section 4.4: An alternative to hard coding

As mentioned previously, the external file would be used to handle dirty data for reporting milestones during an ongoing trial. However, even at DBL, some data issues may persist in spite of DM's best efforts to query and resolve. Perhaps the site was closed prior to the query being issued and has become unresponsive. Perhaps the site cannot resolve a query due to lost contact with the patient. Perhaps the data issue cannot be fully verified with confidence due to the nature of collection. Suppose the site knows that the lab draw for Patient 3004 in Table 4 was prior to first dose, but does not know the exact recorded time. Then, there is no justification for changing the time in the EDC system, but for analysis purposes it can be documented separately that the draw was prior to first dose. Finally, issues within externally provided results (e.g., lab panels or central ECG readings) are difficult to have resolved. Instead of 'hard coding' persistent data issues after DBL within S&P programs (even if well documented), the external file can prescribe the method for handling the data in each case, imported programmatically, and modified without coding changes. This document can accompany the study data submission package (e.g., as an appendix to the clinical study data reviewer's guide) show documented changes to data with reasons.

SECTION 5: CONCLUSION

This paper proposes a data handling technique which we argue improves the efficiency of both data query tracking and statistical programming tasks. The method creates a step of programming to process the transferred raw data for unexpected data issues. Unforeseen data issues identified by downstream programming result in updates to the raw data processing, not the downstream programming. The result is a closed loop of extracting problematic data and creating a mechanism to handle the data issues programmatically, minimizing the need for downstream code maintenance to deal with dirty data. It relies on clear communication channels between DM and S&P groups, particularly across the groups responsible for the raw data processing code and downstream programming (SDTM, ADaM, TLF).

An artifact of this process is an external file which serves as a running log of all potential data issues identified by S&P which impact downstream programming and/or interpretation of results. This file can be communicated back to DM for raw data query, merged into the SDTM database for raw data handling, and document to a third party reviewer any required amendments to the collected data for analysis purposes. Indeed, it can become part of the study data submission package to a regulatory body, at least as part of the clinical study data reviewer's guide.

We believe that our proposed solution can add real benefit to not only minimize the maintenance of downstream programming activities and facilitate smoother reruns on unclean data, but also facilitate efficient query management between S&P and DM groups. And lastly, this approach greatly aids with the handling, review and reporting of unclean and/or anomalous data.

CONTACT INFORMATION

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

Michael S Rimler
GlaxoSmithKline
1250 S. Collegeville Road
Collegeville, Pennsylvania, US, 19426-0989
Email: michael.s.rimler@gsk.com

Jorine Putter
Grünenthal GmbH
Zieglerstr 6
Aachen, 52078, Germany
Email: Jorine.putter@grunenthal.com