**The Basics of Data Management**

By: Tara Reynolds, Christopher Schatschneider and Jessica A. R. Logan

Florida State University

*This work is built from a repository of experience. This work speaks only to quantitative data,*
*and is specifically targeted to research data collection in schools.*

Planning ahead and implementing proper data management will save you time, resources and many headaches. You will get high-quality data that are organized, labeled and accurate. It will also allow your data to be more easily shared, understood and reused by others. Good data management will result in wider dissemination of your work and more citations. Your data must be managed, though, throughout the life cycle of your research: from planning your project to archiving your data.

We will go through four crucial steps that will help you get there: Planning, Data Documentation, Data Entry, and Cleaning Data.

PLANNING A DATABASE

When planning and creating a database for your research, you must consider the data-collection surveys and assessments you are using and also be able to visualize how this data should be stored and structured into tables for analyses. This process is multifaceted and can and should start as soon as possible in the timeline of your research project.

**Databases.** When planning a study, determine first how many different tables/spreadsheets/datasets you will need in order to capture all of the information you will be collecting. If collecting information about children, teachers, and schools, for example, you may need three datasets: one for each type of participant. Also think about how they will relate to one another and how that will be stored from table to table. In a complex project, you may also consider keeping one separate file with demographic information only (student IDs, birthdates, socio-economic status, gender identity). These are pieces of information that you do not expect to change during the course of the project, and that you do not want to store more than one time.

**Identifiers**. In any study, each participant will need an identification or ID. This ID will be linked to that individual’s data throughout the project. The ID is crucial to ensure that statistical analyses such as correlations or regressions can be used to examine how different responses on measures covary across participants. The ID variable is the most critical variable in your dataset.

To ensure that IDs are consistently and appropriately applied, develop a set of rules to govern your ID-creation. There are three critical rules to follow when developing an ID system. First, every individual must have their own unique ID; second, the ID never changes; and third, the ID is always associated with the same person.

The simplest way to create an ID scheme is to give the first individual recruited the ID number 1 (or 100, or 1000) and then count up. It is possible for the ID to include some information (e.g., the year of recruitment, the site of recruitment), however these can make ID rules more difficult to apply. Further, IDs should never include any identifying information (e.g., assigned treatment condition, name of a school). They also must never include any information that may change over time (e.g., the age of the child, the month of data collection). Instead, create additional variables that represent the information you want to store about the data collection.

In addition to these rules, we also suggest that the ID variable should always be formatted as a character variable, even if you are only using numbers in the ID, because many statistical programs try to change numbers to dates and often drop leading zeros. In sum, the key for IDs is to never change your numbering system, never change an ID once it has been assigned, never recycle an ID even if a participant drops from the study, and never have a way for a participant to have two different IDs.

**Variable Names.** All datasets contain variables. Eventually, you (or someone) will write code, using these variables to run your data analyses. You will want to create a variable naming standard, and you should be consistent with this standard throughout your project. There are many ways to do this. Some use the “One-up numbers” system, naming variables from V*1* through V*n* until all variables are named. Although this is consistent and clean, it provides no indication of what data is held in the variable and leaves more room for errors during data cleanup and analyses. Some use the “Question number” for naming variables, for example, Q1, Q2, Q2a, Q2b, etc., connecting the variable to the original questionnaire. This presents the same problems as the One-up method and gets confusing when one Question contains many parts/responses. We have also seen some questionnaires that contain two Question #3’s! So, it can get sketchy.

We tend to use a naming system of 7 characters: the first two characters contain letters that represent the assessment name; the next three represent the subtest (if applicable) then the next two characters represent what type of score it is (RS for raw score, SS for standard score etc.). If you need to number item level data, always put the number at the end of variable name. This system was developed in ancient times when analytic programs required variable names to be 8 characters or less, but we have found that keeping the variable names short and predictable makes data analysis much easier. If you are creating new variable names like this from scratch, run it by a few people first. Make sure that the abbreviation system makes sense to them as well. Another more systematic approach adds prefixes and suffixes onto roots that represent a general category. For example, your root might be MED representing medical history. MOMED might represent the mother’s medical history and FAMED might represent the father’s medical history. Choosing a prefix, root, suffix system requires a lot of planning before setting up your database to establish a list of standard two- or three-letter abbreviations. Always, make sure you keep the naming consistent in your longitudinal projects.

Some further tips on creating variables:

* All variable names must be unique
* The length of the variable must follow the requirements of the data entry and data analysis software packages you have chosen. Many programs do not allow the length to exceed 64 characters. As a rule, shorter is generally better; around 8 and not exceeding 12 characters is a good length to aim for.
* Most programs do not allow you to begin variable names with a digit. Begin with a letter.
* Should not end with a period
* Should not contain spaces
* Should not contain special characters (!, ?, \_)
* Avoid naming variables using keywords that are used during statistical analyses like ALL, AND, BY, EQ, GE, GT, LE, LT, NE, NOT, OR, TO, WITH.
* For item level data, always put the number of the item LAST. This is helpful in data analyses if you are running a sum on 300 variables, you can just say item1-item300 rather than typing out all 300 variable names.
* In addition to naming them, variables have two other main properties: length and type (char, var, date, etc.) Be consistent with the length and type of variables across datasets. Otherwise, merging datasets may become a nightmare!

Once you have your naming system in place, you will want to name and identify all of the variables you will collect. A helpful way to do this is to take a blank assessment packet, highlight every field you want to capture in data entry, and then write the desired variable names on it, along with their length and type.

**Variable Labels.** Many data entry and statistical programs allow you to add a text description to the variable name that describes the content of that variable. This is called a Variable Label. For example, it might contain the item or question number of your data collection instrument, the word being assessed in a vocabulary test, or if it is a calculation field, the way that this number was calculated. The length of a label might be limited, so plan ahead and stay consistent with your labeling method across the project.

**Variable Values.** When translating text to data, it is important to really consider how responses will be recorded (their value) and analyzed from a data standpoint. Also consider the language used in a question and how that ties to your variable name. For example, “Check here if your child does not have siblings.” Typically, people looking at data consider a “checked” box is stored as a “Yes” response or a “1” in data. So, if your variable name is “siblings” and the value is “1” many people will assume ‘yes, this child has siblings’ (which is not the case). You might want to change the question to say, “Check if your child has siblings.” Then a checked value creates siblings =1 which means it is ‘true’ and they do have siblings. If it makes sense on the form to “check” for *not* having a specific value or the papers have already been designed, then maybe consider naming your variable “nosiblings” (then the check mark can still=1 and be true). When responses to multiple items contain the same answer set (e.g., Strongly Agree, Slightly Agree…) keep these variable values consistent across the project. This provides cleaner coding when you get to the data analysis stage.

Tips on creating variable values:

* Usually 1 = Yes, TRUE, correct or present
* Usually 0 = No, FALSE, incorrect or absent
* If Sex or Gender is collected as either “male” or “female,” then instead of using “sex” or “gender” as the name of the variable, use “Male” (or “Female”). Then make the 1’s represent that variable as being present. So, if you named the variable “Male” then make 1 represent males.
* With multiple response items and scales, the variable value in the dataset should contain the same value as the assessment responses to minimize confusion. For example, a multiple-choice question has response options that are numbered 1-5; you will want to store that as 1-5 in your dataset as well. This will allow someone looking at the paper form and also looking at a dataset variable value to understand what is stored there. So, if you want an answer of “none” to be a “0” in your dataset, it should be 0 on the paper also (as people using your data will likely refer to the paper assessment also). Otherwise, you have a paper score of a “2” circled, yet you are storing it as a “1” in your database.
* Stay consistent with scoring your scaled and multiple-choice options. Do you like starting scales at 0 or 1?
* When coding grouped responses (similar from question to question) keep the values that represent the response the same. Ambiguity will cause coding difficulties and problems with the interpretation of the data later on.
* Avoid open-ended text responses when possible. Try to make participants select a predetermined response. When you let participants write/type a response you will get a wide variety of answers for something that should be scored exactly the same way. For example, grade: they could type in 1, 1st, First, Elementary, Early Education, or even have a typo. Make them select or circle “1st Grade”, and then store that value as the numeric value of “1”, not the text.

**Missing Data.** You will need to decide how to handle missing data for every variable in your potential dataset. Missing data could mean a variety of situations, such as the participant did not know the answer, they refused to answer, the question was accidentally skipped, they were unable to answer, you cannot read the assessor’s handwriting, there was an error made by the interviewer and not the participant, they answered outside of the range of allowable responses, the question was not applicable, they were supposed to skip the question and it actually should be blank, etc.

For some projects, you may want to record in your database why the data was missing. One way this can be done is by adding additional variables to the database. However, this can quickly become cumbersome when multiple measures are given. Instead, you may choose to represent missingness with a specific value, with different values representing different missing value codes for the variable in question (see “variable values” section above). Some programs allow for different system-missing values, and you might choose to use that. If you do choose to represent missingness with a selected value, never choose zero as an indicator of missing data. Whatever way you decide to represent missingness, make sure that the missingness decision is documented within your study protocols.

**Planning For Paper Data Collection:**

**Tracking Participation.** It is important to “roster” your study before going out for data collection. Rostering means knowing which participants will be participating in your study. This implies having signed consent forms (if required by your study’s Institutional Review Board (IRB)) and rosters usually obtained from the school.

**Assessment Administration as Packets.** Some research designs require multiple standardized assessments to be administered by a trained professional assessor (project staff). When collecting data with paper forms, assessments are typically given in a specific order (either one set order, or one of a few randomly assigned set orders). When administering measures such as these, there are several opportunities for errors to be introduced. To keep these administration errors minimal, we suggest you develop packets to give to your study participants. Packets are a collection of assessments/questionnaires that will be administered to your participants. Having the answer sheets grouped together into packets eliminates the likelihood of missing an assessment, mixing up which assessment belongs to which student, or delivering assessments in the incorrect order. All packets should have a Student ID or Packet ID pre-printed on each page of the packet.

**Packet IDs.** We encourage people to create something we call “Packet IDs” when creating an assessment packet. Packet IDs are unique IDs associated with each assembled packet, and typically function like a count (the first packet gets ID 1 and so on). Packet IDs can either be preprinted on each page of the packet or printed on labels that can be placed on each page of the packet at any point in time. Packet IDs can then be assigned to a particular student ID (or teacher ID) when administered.

Much like keeping track of the participants in your study, it is also beneficial to track the assessment packets. This has several benefits. First, the Packet ID allows the researcher to minimize the number of times a participant’s name or ID appears on the assessments, which is a requirement for some funding agencies. Second, a Packet ID allows you to easily give an assessment to a student that was not on the roster at the time you printed the assessments (i.e., have an extra stack of assessments with Packet IDs on them, and connect these new students to a specific Packet ID while in the field). Third, if a packet is ruined (have you ever spilled a coffee?), you can grab another packet and associate that Packet ID to the student’s ID. Fourth, if the first page of the assessment packet is a cover page, the link between the packet ID, students name, and students ID can be linked on this page, then torn off and kept in a secure location, which would prevent/minimize data entry personnel from knowing the student’s name.

**Identifying Information and Cover Sheets.** We recommend creating a cover sheet to add on the front of your packet. On this, you will record the Packet ID and any sensitive information such as participant name, date of birth, ethnicity and gender, and the participant’s unique Participant ID. The information recorded here is information that will usually not change throughout the course of the study. Using the cover sheet method allows your team, upon receiving the packet from the field, to tear off the first page of the packet (this cover sheet) and record that the participant’s information has been received, while at the same time immediately separating any identifying information of the participant from their scored data. One designated person would enter this identifying information from the cover sheet into a Demographics Database. You should have one, master, demographic table/file. This is the file that will link the participant’s ID, name, and demographic info to their data. This process prevents the group of people doing data entry from accessing the names of people in the study, thereby limiting the number of people who have access to identifiable data. All remaining pages of the packet will contain labels with the participant’s unique Packet ID, allowing you to enter the full packet’s data at a later point in time in a lab where formal data entry occurs. Another good idea is to record summary data (summary scores) for each assessment on the cover sheet. The benefit of this is that you do, at least, maintain some amount of data if the packet is completely lost, or in the worst-case scenario, if your main dataset of detailed score information somehow crashes. The downside is that it introduces an additional place for errors to occur. Remember, you should always leave the second page (or back page) of the cover sheet blank so that when you rip off the page and store it separately, you will not lose any test data that was collected!

**Tracking Data Collection.** You will want to create a process for tracking everything that needs to happen in your lab and a way to make all of these data collection steps happen properly: where do received forms go, who enters it into the Demographics Database, what happens to the cover sheet, is it kept safe in a locked drawer, where does the packet go when it needs to be data entered, where does it go after data entry, who maintains the list of who is in the study/if they have been assessed, what should you do if you receive data on a student that did not give consent. A project manager should know where the assessment packets are at all times, keeping a paper trail that can be used to check off data as it comes and goes.

DATA DOCUMENTATION

**Protocols.** A protocol is a living document that tracks all decisions made about a project, who made the decision, and why the decision was made. This type of documentation is essential to the running of the project, the ability to make and follow consistent decisions and processes across the life of a project, as well as the ability to describe what was done during the project at a later date (e.g., grant reports; manuscripts). There are several types of decisions where documenting those decisions with a protocol would be useful:

* Variable naming rules
* Inclusionary Criteria: How decisions were made about who to include in the study
* Recruitment procedures and specific ways you might start the communication process with your participants
* When and how follow-up phone calls might be needed
* School requests that differ from school to school or are based on small specifics of classroom rules
* Training of project and lab staff (particularly useful for coding schemes)
* Source material for any measures or scales given (include citations; measurement properties)
* Data entry procedures for the lab (mentioned above)
* How to record missing data
* Data cleaning steps (see data cleaning section)
* What team members should do when in a situation where they don’t know what to do
* Any other items that may be specific to your study

**The Codebook.** The Codebook is also referred to as a Data Dictionary, Data Documentation, Metadata, Data Manual, or Technical Documentation. It is the key that tells everyone what the data in the dataset means. It will be used by all members of your team, including the person performing the data analysis. A codebook also allows others who are not familiar with your project to completely understand your data, cite it, and reuse it responsibly well into the future.

At a minimum, a Codebook should contain:

* All variable names
* Variable labels (could be the exact wording of the question on the assessment)
* Values of the data stored in the variable (which again, when applicable, should be coded consistently across the project).

Other information you might want to include in the Codebook, per variable, are:

* Metadata that was implied during data entry but is not shown in the dataset. This might include the fact that it was collected in one specific state; which wave it was; or that one dataset included teacher’s responses, and another included a school psychologist’s response, yet those values were not stored on the dataset.
* Skipping patterns (the reason certain answers were skipped by the respondent, e.g. “If your response is No, then go to Question #10.”)
* Per-project-unique codes or missing data codes and exactly what they mean.
* A description of a variable that was calculated during or after data entry, including the code used to get that number.

One way to ensure a codebook is high quality is to make a copy of each survey or form administered and superimpose the variable names and values on top of each question. This is a great way to get started.

Some chose to use XML or a searchable PDF, to stay in compliance with the Data Documentation Initiative (DDI), an international metadata standard for data documentation and exchange. Tools that can help create a codebook can be found here: <https://ddialliance.org/training/getting-started-new-content/create-a-codebook>.

Producing a high-quality codebook and keeping it updated as projects progress takes a lot of time, but it is a critical step in ensuring that your information is shared effectively, both within your team and with other researchers. It allows all members of your team to explain results and to share data in open science.

Data Entry

Data that have been collected via paper forms must be translated to a digital database. This translational step is called data entry.

**Choosing The Data Entry Program.** Decide which software you will use for data entry. If you have a quick and simple study, you might want to pick a user-friendly database package like SPSS, EXCEL etc. There are some of the open-source programs like JASP and JAMOVI that are increasing in popularity as well. If your project is large, you may want to create a relational database or use a pre-existing program that allows for data entry such as REDCap, SAS, FileMaker, Microsoft Access, MYSQL, or Oracle. These programs will let you define variable characteristics, create “data entry screens,” incorporate data cleaning which minimizes errors in real time, do sum-to-total checks, incorporate range restrictions, allow you to link multiple datasets with a linking variable, and lock data at the record level. All methods have their respective pros and cons. However, using spreadsheets should be avoided whenever possible. It becomes difficult for the people performing data entry to keep track of what column they are in, and also one “bad sort” can ruin an entire data set. Spreadsheets should be reserved for very small studies where if one had to re-enter all the data, it would not be devastating.

**Creating The Data Entry Screens.** A data entry screen is the interface between the hard copy of the data and the digital database. It is a user interface, designed to minimize the cognitive load on the data enterer and therefore minimize errors during data entry. When you create data entry screens, you will want to make them look as similar to the packets as possible. This will not only save time for data entry personnel but will help them stay focused on what they are entering, rather than focusing on where they are in the program and feeling lost between the paper and the screen.

Within each form, the flow of the data entry should follow the logical order of each form. A paper form with 20 items (#1-#20) should allow the data enterer to type in the response to the questions in the order that they appear on the form. This is called “Tab order” and dramatically improves the data form entry process.

If a project contains multiple assessments, the order of the assessments on your data entry screens should follow the order of the paper packet in the data entry person’s hand. When they have finished entering one assessment, and turn the page, your “next” button on the data entry screen should advance them to the next assessment they are looking at on paper. Stay in synch with them. This helps them save time and stay focused.

The data entry screen provides an excellent opportunity to prevent data entry errors. Define range limits on your data where possible (for example, the score must be a value between 0 and 30; the birthday and/or assessment date must be between certain dates). Do not allow a required field to have a null value. Do not allow entry into certain fields if the criteria of the participant should not allow them to answer that specific question.

After creating the data entry program, you should sit with the data entry team and watch them enter a few forms. You will find there might be misunderstandings about the data, or that there is something you can tweak on the screen to make the process easier for them. Test it with them before going “live.” After they enter a few forms, you will also want to check the data against the participant’s paper form to make sure all data was translated correctly; and then, run the data by the PI and the Data Analyst to make sure it is what they were expecting.

You will want to make sure that your department/university has a file server. This allows multiple computers to access the same data files, and it is easy to set a process that backs up your data. And most importantly, MAKE SURE YOUR DATA IS BEING BACKED UP THROUGHOUT THE ENTIRE PROCESS!

**Calculated fields**. The data entry team may be entering item level and sum score information that is calculated by a human, in the field. You will probably want to create a system-calculated field using a formula and compare your calculation to the assessor’s calculation. This code calculation happens AFTER data entry, on the back end. (You do not want it calculated on the data entry person’s screen, as they should not be looking at calculations and comparing them during this time. They should only be doing data entry.) If the two calculated scores do not match, then send it back to the data entry team to find the problem.

**Quality Assurance.** Ensure that you have some kind of data entry quality checks. We recommend all data get entered twice. That is, have a primary and a secondary dataset that are supposed to contain the same information. Then these two datasets should be compared for discrepancies and any data that do not match should be flagged and investigated. Many data entry programs do this step for you; otherwise, discrepancies can be reconciled either in a third dataset or through hard coding (syntax).

**Longitudinal Considerations.** Designing a data entry program for longitudinal databases can take on many forms. We recommend creating different files/sheets/tables for different waves of data. Use the univariate format – which means use the same variable names for the data collected over multiple time points. Do not add numbers to variables to designate “wave” or end them with an F or S to represent fall and spring. Those can be added later programmatically, and the univariate layout makes data cleaning easier. In the “wave” datasets, only enter what you collected at that wave (plus ID and date of collection). Do not add anything else. Maintain a separate file with demographic information for each subject (this is the only place that a subject name can appear). All other datasets should only use IDs.

**Beginning the Data Entry Process in the Lab.** You’ll need to set up data entry stations or a lab where people can simply sit down and enter data. The data should stay at a station and should never follow a person if they need to walk away or go ask someone a question.

The project manager must maintain some kind of master list that tells her/him what is expected at the end of each wave in terms of which participants were tested. This can be created from consent forms or school rosters and can be easily maintained in EXCEL. At all times, this person should have a master list of all students and know who is in the study. This master list might be from the Demographic Database and contain contact information on the family.

When a packet arrives from the field, it should be immediately checked by someone for accuracy (missing data, miscalculations, missing labels). Checking this right away can catch administration errors that can greatly impact your data. Checking the protocols at the front end will reduce significant cleaning time at the back end.

If the packet looks good, immediately log it into the Demographic Database. The database should only exist in one place. It will contain the student ID, packet ID, and any other sensitive information. The benefit of doing this is the safekeeping of your data. One trusted person looks at it, as you will want as few eyes as possible seeing any sensitive, identifying information. The process will also help your team keep track of which students still need to be tested at the schools (because you have already rostered; and you know who needs to be tested) or if a form has been misplaced. Many students miss kids/a classroom, and this will help you get more of your data on time.

There should be an easy system in place, where each person, without effort or documentation, knows what needs to be entered. You might want to have file cabinets clearly labeled, for example, “Needs double entry.” The data entry team should never have to figure out which packets have been entered, and which have not, by pulling up a spreadsheet. This also will prevent simple errors like a packet being entered twice.

Data entry should be as thoughtless as possible. No math should be done in the middle of data entry. Scores should have been calculated in the field, but if not, perhaps have a place in your filing cabinet system for “needs scores,” before “Needs to be entered.”

People should only have access to their data entry program module. If your data entry program has other studies in it, they should not be able to see or access those. Once logged in to their study, they should only have data access to the functions that are available only to their role in the project (first entry, secondary, reviewer, etc.) and no one else’s roles. Make sure your system, as defined by your research grant, is in place and being followed.

The PI and Data Analyst should develop Data Entry “Rules” for ambiguous cases. The data entry team should be well trained on this, and have a written, easy Rule Book at each data entry station that they can refer to while entering data. Cases might include what to do when: the form or question has no answer, there are blanks, the respondent did not follow directions, or the respondent circled two options when only one was allowed. The key, as always, is to be consistent!

When the data entry team is completely finished with their entries, they will notify the Data Manager, who will extract the data from the data entry program.

**Data Collection With Electronic Systems.** Instead of paper assessments and a team of people entering the responses from paper into a database, many researchers may want to use electronic web-based systems to collect data, such as Qualtrics and REDCap. We have some specifics tips and tricks. Online systems for delivering surveys to participants often name variables in a confusing way. In Qualtrics for example, variable names are often Q1, Q2 etc., but the numbers relate to the order in which the questions were made, not the order in which they appear on the survey. Further, the values assigned to categorical variables are also not intuitive and can even change per question. When creating surveys, most programs will have an option to change the names and default values for the created variables. Examine options to ensure these are in place before ever administering one survey. Ensuring the variables are named how you want them and that the values assigned are consistent will save you a lot of time later.

If we have caught you too late, and the survey has already been administered, then make sure you refer to your documentation about the surveys to create code to change those names post hoc. Using code will help keep transformations transparent, and repeatable, when or if you need to make the same changes again.

A second thing to think about is pulling the data from the server to your computer. Often it will look very different than what you had expected. Qualtrics, for example, adds many columns to the data that you had not programmed. These include longitudes and latitudes, IP addresses and other data that could potentially identify the respondent. It is a good idea to do a couple of practice data collections with your platform and inspect the datafile for these unexpected things before starting actual collection. You can then anticipate cleaning and possibly make changes to the survey to better suit your needs.

CLEANING DATA

**Before you start.** As you clean your data, remember to always keep a trail of everything you do. Always work in a copy of the dataset, maintaining the original dataset in its raw state. Always back up your data and documentation as you work.

All data management code should be saved and annotated as you go. Explain what the code is doing, what you are looking for if running a query, what kind of a check you are running, or why you are doing this (for example: “Required by IRB to …. “). Anyone who looks at your code should be able to read your annotations at every level and understand what is happening and why. They should not have to read your programming code to figure out what you were doing or what your intentions were.

Example: Your assessment has a rule that the assessor records the highest item number the child answered. So, in your annotation, you might note: /\*This grabs the highest item number the assessor said the child answered (for example, Question #85) and makes sure no questions were scored beyond that (if child answered questions higher than question 85, then you want to flag it) and that all previous questions before Q85 were indeed answered.\*/

Second example: You need to remove certain rows from the dataset before analyzing the data. Your annotation should explain that you are indeed deleting a participant and why: /\*Child was removed from study, due to moving schools before testing began/not having enough data... \*/

Not only will these annotations help the next person that might be looking at your data process, but these detailed notes will also help you when you revisit your data multiple times during the study. It prevents you from overlooking decisions made about the study, and it allows you to quickly run through all pre-established requirements if you need to run the file again for some reason in the future. (Example: two more participants were added to the data entry program after the project was “over,” and all data must be re-extracted, and code must be re-run).

**Structure.** You will now create and manage the working (aka transactional) and master datasets. Transactional datasets are datasets that contain cleaned data and any derived data such as computed sum scores, transformed variables, and potentially norm-referenced scores if they were calculated via code. To begin, structure your data into the file layouts that work for your project. You may need to join tables or rearrange your data into long or wide datasets. Get the data files organized in a way that fits your research needs.

Make sure you are only dealing with the rows of data that have been double-entered and verified. Data entry programs often maintain the first entry, second entry, and finalized entry. Make sure your working dataset only contains the finalized record for that participant.

The participant ID, Study ID, and any identifying variables should be located at the front of the dataset, in the first few variables.

**Checking the Data.** You will begin by checking the data, to make sure everything looks correct. Mistakes can still be present after double entry. You will want to write code to check for obvious mistakes. You will also want to check your dataset, via code, for project-specific rules (e.g., no one over age 5 should be in this dataset). This checking part of the process requires fully understanding the project, the protocols, the assessments, the data, the study requirements, and the intentions of the Data Analyst. You will have to define these code steps on your own, based on the specifics of your project.

Examples of some general checks we run include:

* The numbers of participants you have match the number of participants given to you by the project manager
* Every ID that is in the table should have been assessed (ties back to master list), to make sure you have no unexpected students and/or to make sure no one is missing
* Eyeballing your data – surprisingly, this is quite helpful
* Is the variable in the correct format (char or numeric)?
* Sum-to-total checks
* Sum scores on the summary sheet match the sum scores on the score sheets
* Range checks
* Duplicate ID entries, duplicate form entries
* Basal and Ceiling rules
* Validity checks
* Run descriptives on key variables/calculated scores
* Run frequency codes just to get a fresh look at your data. You might find a value that just seems off
* Run Consistency Checks that are particular to your study and based on your extensive knowledge of the study.
* Double checking all calculated fields like “Total Scores,” whether it was entered by a human or calculated in the data entry program
* Are the scores expected for that particular wave and timepoint present; and vice versa, do you have scores that were not expected for that participant/timepoint
* Have participants been assessed at every timepoint required of them by the protocol (pretest, posttest, delayed posttest, etc.)
* Do the means make sense across ages/grades/timepoints
* Do you need to find incomplete records?
* You are only looking at rows that have been double entered and verified
* Inconsistent responses per participant. Or perhaps, has the respondent selected the same response for every item?
* Check project-specific protocols. There will be a lot of these. This might include things like “no one was given a PPVT assessment in the spring.” So, make sure there is no data for that. Check that all project-specific requirements have been met.
* Did an assessor mark that this assessment “was not administered” yet have data for it. Go back and have the data entry team research it.
* Ensure that the variables and their values match those in the codebook, you do not want to have a “5” response in your study that is not explained in your codebook.

Sidenote: We recommend running some of these code checks during the data entry process here and there. You never know what kind of issue might pop up, and it is better to fix it earlier in the process rather than later.

**Fixing Problems.** Create a system for tracking issues you find within the data and their resolutions. When mistakes are found, often you will pull the protocols and conduct a “hard target” search. This means retrieving the original paper assessment to make the corrections. How you make the correction might depend on the extent of the problem and how it made it to the final dataset. For example, you might be able to have the data entry person correct the error on their end. If, however, you feel like the error should be corrected via hard code within your working dataset (e.g., a birthdate was incorrect and needs to be corrected), then you will begin the process of making a “transactional dataset.”

This brings us to a very important principle. CHANGES TO DATA ALWAYS FLOW FORWARD. What does that mean? That means your primary datasets should never change. Any reconciliation occurs in a separate, “transactional” dataset. Always work from a copy, and keep extra copies of clean data in other locations. You should never rewrite the original files produced during data entry. Why do we recommend this? If something goes wrong or mistakes are made in the data cleaning process, you will always have the original datasets to fall back on. Using a transactional dataset and code to correct data ensures you can track all changes made from entry to final dataset. DATA FLOWS FORWARD, NOT BACKWARD. You should always be able to go back to the originally entered data and be able to see where values got changed and why they were changed.

Within your transactional dataset, you might also calculate or add values like standard scores, percentile ranks, confidence bands, and normative scores. For assessments that convert raw scores to percentile scores, you might have had a person manually add a score during data entry using a table to match each student’s criteria (age/grade/score) to the standard score and then enter that into the system during data entry. If not, however, you might have decided to instead use lookup tables to create those values. With lookup tables, you create a separate dataset with all possible scores and their corresponding percentile/standard scores. You then join each participant to this table to retrieve their score, based on the assessment criteria. It is more work on the front end, but it is more error free and works out nicely when dealing with larger amounts of data. Some assessments already have published norming tables out there that you can use.

All changes to data need to happen in one data file. You should not have multiple versions. If you need to add 800 corrections or new variables, go ahead and do it, but do it all in ONE dataset.

Remember to document where your data is backed up and stored. Multiple versions of final datasets ensure their safety. Documentation should include how many copies you will keep and how you will synchronize them. It is a good practice to also include security issues such as passwords, encryption, power supply or disaster backup, and virus protections.

Once you have made all data management changes, you will have one master dataset that will be ready for analysis. This dataset will hold everything and can be quite large.

**Analysis Dataset.** Datasets that can be used for analysis can also be created. These datasets are typically a subset of the variables or subjects from the master dataset. These datasets should always be created from the master datasets, but the master datasets should never be changed. Once data analysis has been completed and the manuscript sent off for review, the analysis dataset needs to be “frozen.” That is, no additional changes should be made to it, and the datasets should no longer be reconstructed from the original master datasets. This ensures that you will always be able to replicate the results of the analyses reported in any paper.

SUMMARY

In summary, many potential errors can be avoided by carefully planning and discussing all the different portions of the data life cycle before a data collection begins. It is a great idea to: (1.) define your data management guidelines during the design stages of your research study, (2.) take action during the study to ensure those plans are continually being revisited, applied and discussed, and (3.) complete all documentation about your data and store it with your archived data so that it can be understood and reused by others. These actions will likely fill funding agency requirements and ensure more citations for your publications. If you have any tips you’d like to share with us, feel free to email treynolds@fcrr.org.