**ETL (Extract, Transform, Load)**

​As discussed here, the data exported from GemsTracker contains two files per survey:

1. A file with the respondent data with al the data that each subject filled in. Each time someone fills in the form, a row is added. So one subject can have multiple row
2. A codebook of the survey, defining the variable names and the scoring.

Since measurement tracks in Gemstracker generally contain multiple questionnaires (PROMS or clinician-reported forms), data from all questionnaires have to be combined in a comprehensible format and coupled on a subject level so that you know when scores from different questionnaires are from the same subject.

In an ETL script, all exports of all questionnaires in a measurement tracks are imported into R, preprocessed, and after some basic cleaning stored in separate files with each questionnaire or form in a list.

The ETL of the HandWristStudy group is even more extensive since it combines data from 8 different measurements tracks (e.g., a track for a patient undergoing thumb surgery and a track for a patient with a neuropathy).

Although ETL’s can be defined in different ways, below, we explain how an ETL can be made for a very simple case of a measurement track with only two questionnaires with example data from a few simulated patients.

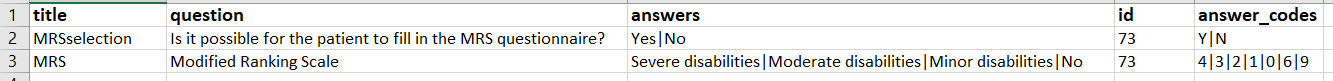
In the R repository, in the folder scripts/ETL, you can find an r script called example\_ETL.R that follows the steps below.

**Matching codebooks**

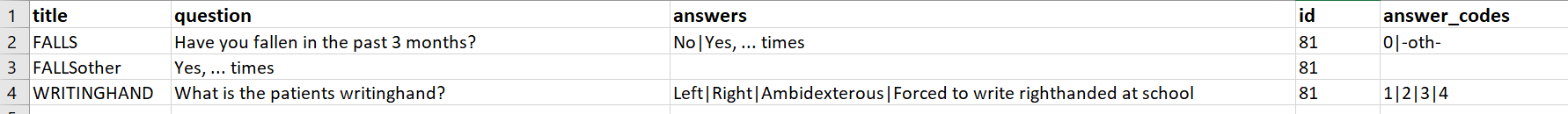
In the first step of the ETL, the script checks whether the questions in each survey have remained the same. This check is necessary since sometimes changes are made to the data collection and this needs to be detected. Therefore, it is a first before you can run the rest of the ETL script.

To check the matching, the codebooks of all surveys are loaded and combined into one large codebook. To make sure all questionIDs are unique, the title of the question is combined with the surveyID into the variable *rowID* (line 36-40), resulting in the following:

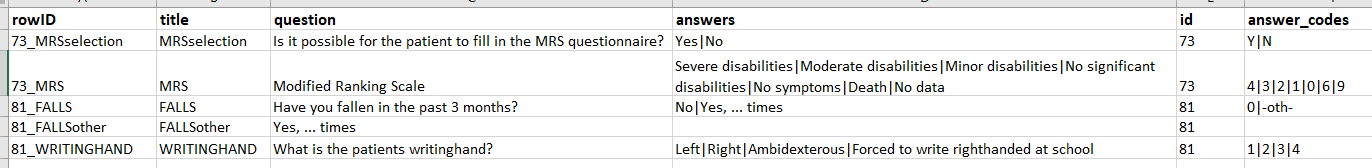
**CODEBOOK SURVEY 73 BEFORE PROCESSING**



**CODEBOOK SURVEY 81 BEFORE PROCESSING**



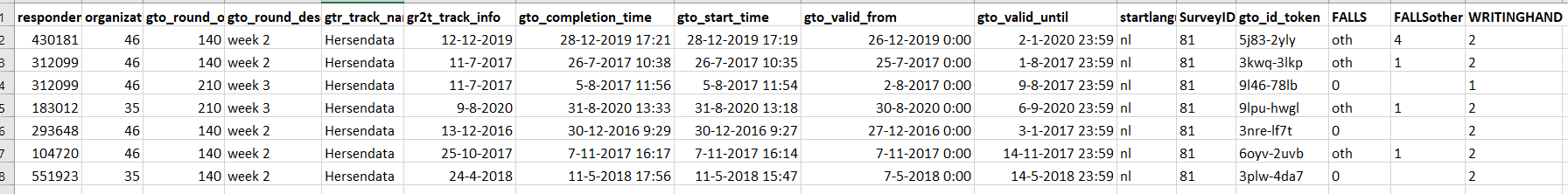
**MERGED AND PROCESSED CODEBOOK**

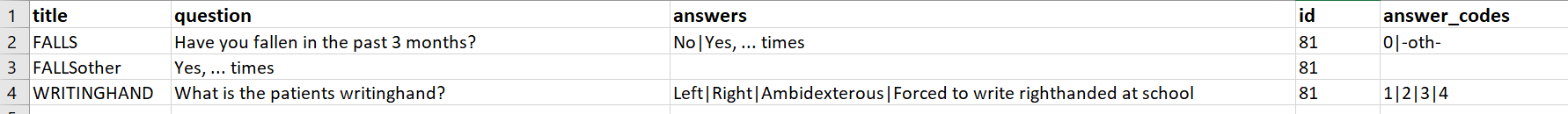


The merged codebook of the current export should have the exact same rowIDs as previous exports. Therefore, it is compared to a reference codebook used to develop the ETL. In the first place, this check is done to make sure no surveys were forgotten during the export. Furthermore, this check also gives insight to whether questions are added or removed, which might affect the ETL script. If there is a difference in one of the surveys compared to the previous version, a warning comes up stating there is a mismatch with the codebook, plus information about the missing question or added question (line 76-78). In case no surveys have changed, the ETL script will continue to load the respondent data (line 58-59).

**Recoding raw data**

When exporting GemsTracker data, all surveys are stored as separate Excel files. The raw data exported from GemsTracker uses the answer-codes in LimeSurvey to store all the answers. However, these answer-codes are not always easy to interpret. Before recoding the raw data, it often has a structure like shown below. The left part of the data contains all the information about the measurement tracks (red). The middle part shows the token information (green, describing date, time and technical details about when the data were collected) and on the right, you will find the answers to the questions of the given survey (blue), in this case, survey 81.

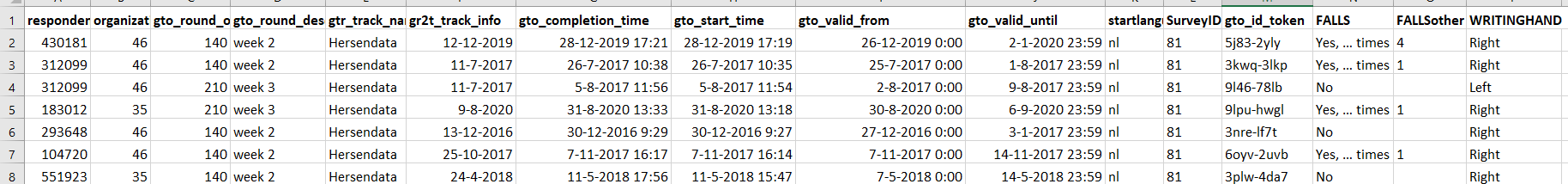




The codebooks are used to translate all the information in the blue part of the data into something more interpretable. Within the ETL, the answer codes for each question in each survey are coupled to the answers as they were presented to the respondents. For example, when someone is lefthanded, they have filled in “Left” for the question WRITINGHAND, while this was stored as “1” in the raw data.

Questions in which no answer options are given (text field or numerical answers) do not have to be recoded, which makes that the answer\_codes and answers for these questions are empty in the codebooks. For Survey 81, this is the case for the question FALLSother, where a number can be filled in when the previous question is answered with “Yes”. In the code, this can be found in line 102 (for survey 73) and in line 119 (for survey 81), where empty fields of the codebook are skipped.

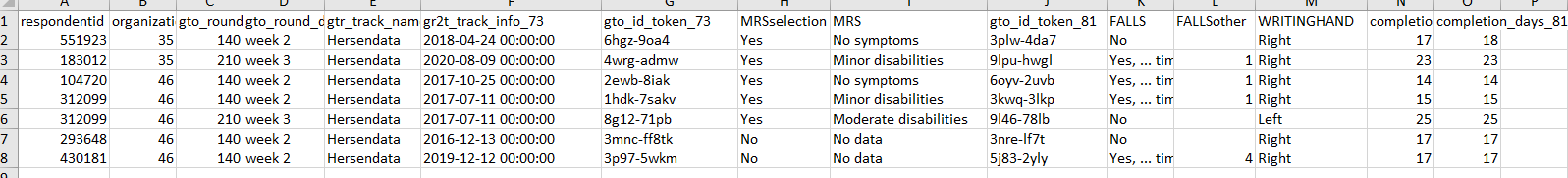
For all the answers that do have to be recoded, factors are created with the answer\_codes as levels and the answers as labels of the factor (line 103-107 and line 120-124). After recoding one survey, the data would look like this, where the categorical answers are stored as factors:



**Merging multiple surveys**

Besides decoding the information for each survey, the ETL is also responsible for merging all patient information stored in multiple surveys over multiple rounds (e.g., baseline and follow-up) and even in multiple tracks (first in the track for non-surgical treatment and then in the track for surgical treatment).

To combine all data from one measurement track, surveys are merged based on respondent ID, respondent track ID and round description (line 131). After this, one row represents all information from one patient from one measurement round. When a variable name is present in multiple surveys, the surveyID is added to the name of that variable (line 132). For example, the variable gto\_id\_token was present in both survey 73 and survey 81; therefore, the new names are gto\_id\_token\_73 and gto\_id\_token\_81. In the end, you will have one dataframe containing all the information:



**Merging multiple surveys**

This concludes the basic processing of this example ETL.

However, in the concept of a data pipeline, it is generally wise to do all corrections as close as possible to the source. Since the source is Gemstracker, any correction that you can do in Gemstracker is better than later. Since the next step is the ETL, it can be wise to use the ETL for all kinds of general data cleaning and calculations. For example, it can be a good place to calculate the sum score for questionnaires or calculate BMI from length and weight. This way, all researchers using the output use the same calculations and correct them when needed.