Institute of Education

UCL

Millennium Cohort Study

Data Handling Guide

with syntax in R, STATA and SPSS

August 2020

CENTRE FOR LONGITUDINAL STUDIES

UKRI Economic and Social Research Council

Contact

Questions and feedback about this user guide should be sent to clsfeedback@ucl.ac.uk.

How to cite this guide

Agalioti-Sgompou Vilma & Jon Johnson. (2020) *Millennium Cohort Study Data Handling Guide with syntax in R, STATA and SPSS*. London: UCL Centre for Longitudinal Studies

This guide was first published in July 2020 by the UCL Centre for Longitudinal Studies.

UCL Institute of Education

University College London

20 Bedford Way

London WC1H 0AL

www.cls.ucl.ac.uk

The UCL Centre for Longitudinal Studies (CLS) is an Economic and Social Research Council (ESRC) Resource Centre based at the UCL Institution of Education (IOE), University College London. It manages four internationally-renowned cohort studies: the 1958 National Child Development Study, the 1970 British Cohort Study, Next Steps, and the Millennium Cohort Study. For more information, visit www.cls.ucl.ac.uk.

This document is available in alternative formats. Please contact the Centre for Longitudinal Studies.

tel: +44 (0)20 7612 6875

email: clsfeedback@uck.ac.uk

Contents

Acknowledgements

We would like to thank Aida Sanchez, Emla Fitzsimmons and Vanessa Moulton for their feedback when producing this user guide.

Finally, we would like to thank researchers who used early versions of the guide and provided us with their insight.

About the Millennium Cohort Study

The Millennium Cohort Study (MCS) is a longitudinal birth cohort study, following a nationally representative sample of approximately 19,000 people born in the UK at the turn of the century.

Through the study, we have captured rich information about the different aspects of cohort members' lives, from birth to childhood and adolescence, and we are continuing to keep up with them now they are adults.

As a multidisciplinary study, MCS is used by researchers working in a wide range of fields. Findings from MCS have influenced policy at the highest level, and today the study remains a vital source of evidence on the major issues affecting young people's lives.

Sweep	Fieldwork / data collection starting year	Cohort Members' average age
MCS 1	2001	9 months old
MCS 2	2004	3 years old
MCS 3	2006	5 years old
MCS 4	2008	7 years old
MCS 5	2012	11 years old
MCS 6	2015	14 years old
MCS 7	2018	17 years old

About this user guide

The Data Handling guide aims at helping researchers use the data of the Millennium Cohort Study to its fullest potential. We focus on explaining the data structures of MCS and on providing solutions with different data handling strategies.

The structure of the guide per chapter is the following:

Chapter 1 – Overview of the data structure and the identifiers of the datasets of the MCS.

Chapter 2 - Explanation of how to use dataset of different structures separately but also how to merge between sweeps.

Chapter 3 - Detailed information on why restructuring may be needed and how to proceed with it, including walkthrough on merging datasets of different structures.

Chapter 4 - Finally, this guide provides the syntax that has been used for each chapter and sections. The syntax is in SPSS Syntax, R and STATA.

Important notes about this user guide:

- This user guide does not replace any of the User Guides of each sweep of MCS. Those guides contain specific information on survey design, questionnaire, survey outcomes and detailed information on the derived variables.
- The example codes are provided in the Appendices and they can be used for a hands-on experience with the data.

We hope you enjoy working with the Millennium Cohort Study!

Abbreviations & shortcuts housekeeping

There are some abbreviations used in the text to make text more condensed that are important to familiarise with:

- MCS - Millennium Cohort Study
- CM - Cohort Member
- CMs - Cohort Members
- CAPI - Computer Assisted Personal Interview
- CAPI name/code - the 4-character code of a question as it appears in the questionnaire

In this guide we use the term **parent** and **carer** interchangeably.

1. Data structures and identifiers of MCS

1.1 From the survey design to the data structures

This section explains how the data of the Millennium Cohort Study are structured. It also contains explanation of how the various data structures of MCS are generated by the distinctive design of the survey interview and the questionnaire.

1.2 Variations in the family structures

A typical interview setting of MCS includes the Main carer of the Cohort Member(s) (usually mentioned as Main), the Partner of the Main carer (usually mentioned as Partner) and the Cohort Member(s).

Family structures vary and MCS captures this variation:

- Some families had multiple births: twins and triplets of a family are all eligible to participate as Cohort Members in MCS.
- Sometimes the Partner respondent although s/he lives in the household s/he is not available for the interview and the Main respondent provides information about him/her (Proxy Partner module).
- Sometimes there is no person eligible for the Partner interview and there is only a Main respondent for that family.
- The largest number of individuals eligible for interview in one family in MCS is 5: 3 children (if triplets) and 2 carers/parents.

The key to using the data effectively is the questionnaire. The questionnaire provides information on whether:

- the question is addressed to **Main only** or **Main and Partner** or **Partner only**
- the question is about themselves and the family or it is about the Cohort Member(s), and,
- the questions get repeated (in a loop) for each Cohort Member of the household.

1.3 Distribution of variables between dataset types

The picture below shows how different questions collect information that varies in terms who the respondent is and whom the question is about (parents about themselves or about the cohort member).

The focus is on who answers the question and about whom.

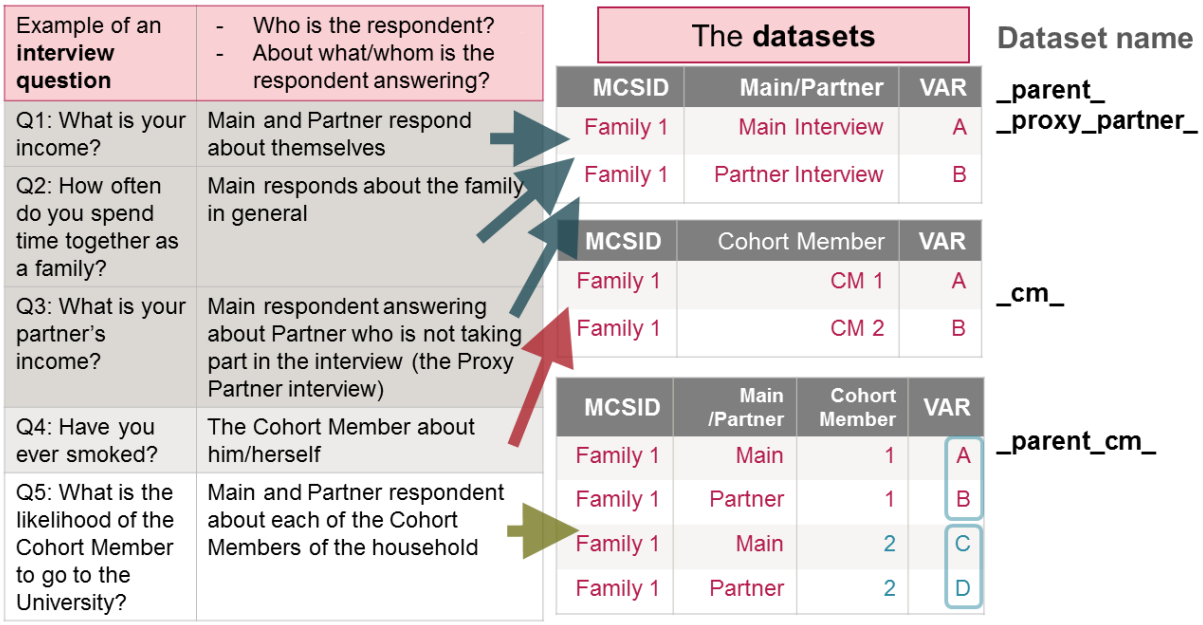

Figure 1: Interview setting -> Questionnaire -> Datasets

The datasets are available in 4 structures - or levels - depending on the information they include:

- Family level (one row per family)
- Parent level (one row for Main and one for Partner respondent)
- Cohort Member level (one row for each of the Cohort Members)
- Parent - Cohort Member level (one row for each parent respondent and then one row for each of the Cohort Members)

This rule of distributing variables applies to the derived variable datasets too.

1.4 Dataset types

The structure of the data is part of the dataset title to make it easy for the data user to know the level(s) inside the dataset before accessing it.

Family level dataset _family_: These datasets contain information about the family as one entity. For example, in the mcs*_family_derived dataset, one can find information about the family type, the total number of Cohort Members in the household, etc. The mcs_longitudinal_family_file that contains the weights is also on the family level.

Parent level dataset _parent_: If the question is addressed towards the parent and it is about the family or about him/herself. This is regardless if the question is to be addressed to Main respondent only or to Main and Partner respondents), then the variable that corresponds to that question is in the _parent_ level dataset. For example, the variables of parental income questions can be found in the _parent_ level dataset. The Parent level dataset includes the Household Module which contains demographic information about the family.

The Proxy Partner interview occurs when there is a person eligible (ELIG, see identifiers) for the Partner interview but s/he does not participate as Partner (RESP, see identifiers). In these cases the Main is asked some questions about the Partner (hence, Proxy Partner). The _proxy_partner_ dataset has similar structure to the _parent_ level dataset (one row per partner).

Cohort Member level dataset _cm_: If the respondent is the child answering questions or providing information about her/himself, then this is included in _cm_ level dataset. Physical measurements of the Cohort Members and cognitive assessments are in _cm_ level datasets. From sweep 4 onwards, the CMs get interviewed and this information can be found in the _cm_interview dataset.

Parent - Cohort Member level dataset _parent_cm_: Many questions in the early sweeps of MCS ask for information about each Cohort Member from the parent(s).

Dataset structures

Variables are allocated to datasets according to the type of information they hold. The questionnaire specification (to whom the question is asked and whether it loops over each CM) determines the type of information held. For example, if a question is answered by the Main or Partner respondent about themselves (Parent level dataset) or answered by the Main/Partner about each of the Cohort Members (Parent - CM level dataset). The same frame applies to derived variable datasets.

family level dataset

MCSID	VAR
Family identifier	
Family 1	A
Family 2	B
Family 3	C

_cm_ level dataset

MCSID	Cohort Member	VAR
Family identifier	Index of the Cohort Member (CNUM)	
Family 1	1	A
Family 2	1	B
Family 2	2	C
Family 3	1	D

parent level dataset

MCSID	Main/Partner	VAR
Family identifier	Variables on Main/Partner respondent (ELIG, RESP, PNUM)	
Family 1	Main Interview	A
Family 1	Partner Interview	B
Family 2	Main Interview	C
Family 2	Partner Interview	D
Family 3	Main Interview	E

_parent_cm_ level dataset

MCSID	Main/Partner	Cohort Member	VAR
Family identifier	Variables on Main/Partner respondent (ELIG, RESP, PNUM)	Index of the Cohort Member (CNUM)	
Family 1	Main Interview	1	A
Family 1	Partner Interview	1	B
Family 2	Main Interview	1	C
Family 2	Partner Interview	1	D
Family 2	Main Interview	2	E
Family 2	Partner Interview	2	F
Family 3	Main Interview	1	G

Figure 2: Overview of dataset structures and the identifiers they contain

The structure of this dataset type reflects the information collected by the survey questionnaire. For example, a question that is asking each of the parents **whether the CM cries at night** gets repeated for each Cohort Member.

This happens if the question about each CM has been addressed to both parent respondents (Main/Partner) or to the Main respondent only. The dataset includes identification of the respondent (Main/Partner) and the Cohort Member the question is asked about (Cohort Member number).

The **household grid** is on person level (PNUM / CNUM) since there is one row for each person of the household regardless if they were selected for an interview (for example, CM's siblings, grandparents, other relatives or non-relatives). The household grid is a source of information about the key respondents of the interview (Main/Partner/Cohort Members) as well as the rest of the members of the household that do not participate during the survey interview.

In certain sweeps, there is an **older sibling** dataset where there is a row per older sibling that has participated. Also, there have been teacher surveys of the Cohort Members which have a _cm_ structure since the teacher provides information about the CMs.

The **_derived** datasets contain derived information about the demographic characteristics and socio-economic status of the household and its members. The _derived datasets can be found in 3 different structures:

- cm_derived with information about each Cohort Member (one row per CM)
- parent_derived with information about each Main or Partner respondent (one row per parent)
- family_derived with information about the family and the interview (one row per family)

Information about each variable of these datasets and how they have been derived can be found in the respective User Guide of the derived variables of each sweep.

1.5 Identifiers: MCSID, PNUM, CNUM, ELIG, RESP

The format of the different dataset types, require different identifiers that help the user manage the data for each family, parent/carer respondent (within the family), Cohort Member (within the family) and person (within the family). This section describes the identifiers and how they can be used.

The key identifiers of MCS are:

- **MCSID** is a family / household anonymised identifier and it is the same for all of its members, Cohort Members and parents per family.
- **CNUM** is Cohort Member number within a family. Namely, the CNUM in ascending order standing from 1 indicates the first Cohort Member within a family. The majority of the Cohort Members have CNUM=1, however, in families with twins and triplets the second and third Cohort Member have CNUM=2 and CNUM=3 respectively.
- **PNUM** is Person number for the individuals living in the household apart from the Cohort Member(s). This includes grandparents, siblings of the Cohort

Members, parents, etc. The PNUM is given in an ascending order starting from 1 randomly to each person that lives in the household that the Cohort Member lives or has lived (in a previous sweep). This means that a grandparent may be PNUM=1 and if s/he moves out in a later sweep s/he keeps the same PNUM. If another person moves in to the household or a sibling is born will receive the next available PNUM.

- **ELIG** provides the information on whether the individual has been eligible for the role of the Main or Partner respondent. For example, the mother of the Cohort Member may have PNUM=2 and the father PNUM=3, however, only the mother may have been eligible to be interviewed for the survey (ELIG=1) and not the father (ELIG=4).
- **RESP** marks whether the person selected at ELIG has participated in the interview or not (e.g. refusal). This variable is the outcome of the ELIG.

PNUM and CNUM are permanent cross-sweep identifiers. Namely, they do not change between sweeps and the person or CM holds this number for the rest of the survey. Moreover, the PNUM gets assigned at random. It is likely that the parents of the CM have a low PNUM as they have been present at Sweep 1 or 2 but this is not always the case.

ELIG and RESP are sweep-specific identifiers. Namely, they mark whether the person with a specific PNUM has been selected to participate in the interview and provide information about the Cohort Member(s) in this specific sweep (as Main or Partner respondent). The individuals eligible to respond to a survey can change from the one sweep to the other. For this reason PNUM is available in all datasets where there is a parent respondent like the _parent_ level dataset and the _parent_cm_ level dataset.

Each dataset depending on its type and structure contains a different set of identifiers. For example:

- a dataset on the CM level _cm_, such as the self-completion questionnaire of the Cohort Member, contains only MCSID and CNUM and not any parent identifiers (PNUM, ELIG, RESP) since these are irrelevant,
- a dataset on the parent level _parent_, contains MSCID, PNUM, ELIG and RESP but does not contain CNUM,
- a dataset on the _family_ level contains only the MCSID which is the family identifier,
- figure 3 provides an example of the _parent_cm_ level dataset that requires all the identifiers. This is because within a family (MCSID) a parent (PNUM, ELIG, RESP) provides information about each of the Cohort Members of the family (CNUM).

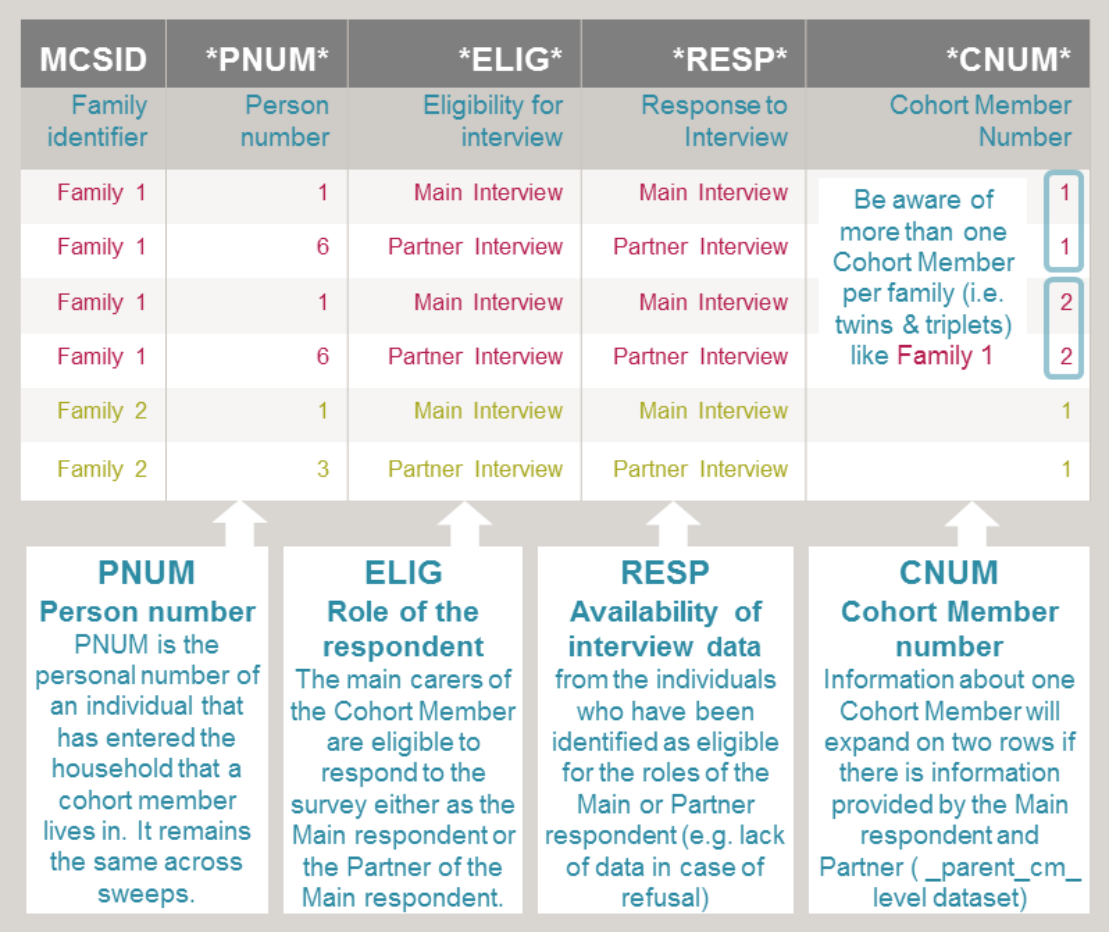

Figure 3: Example of how key identifiers look like in a _parent_cm_ dataset

1.6 Parent or Carer respondent and the distinction between PNUM and ELIG/RESP

The existence of PNUM, ELIG and RESP is a design characteristic of MCS that provides the data with an important advantage: there is a person providing information about the CM even if the natural parent does not live in the same household with the Cohort Member or is not available to respond.

When working with datasets that are on the _parent_ , _parent_cm_ and _proxy_partner_ levels it is important to consider that the parent respondent (Main/Partner) may change between sweeps.

The possible family structure changes vary. The most common one is for a natural parent not to be in the household (for example, divorced, moved out, passed away) and another carer (if available) of the Cohort Member, like a step parent or grand parent, becomes eligible to participate in the survey interview.

There is a different PNUM for the person who moved out and for the new person who moved in. For example, the natural parent may have a PNUM=3 and the step parent (or grandparent) may have a PNUM=5.

Although the PNUM will remain the same in the subsequent sweeps for both (in the household grid), ELIG contains their eligibility for interview for the role of the Main or Partner of the Main carer a different person in a specific sweep. Eligibility depends on who has been a resident in the same household as the Cohort Member at the time of the interview.

1.6.1 Merging between sweeps: focusing on data availability (ELIG)

For some research projects the focus may be on information provided by the Main/Partner respondent about the Cohort Member(s) regardless of the fact that the Main/Partner respondent may change between sweeps. For example, in the one sweep the Partner respondent may be a natural parent whereas in another a step

parent. In this case, a data user may prefer to merge the data between sweeps using MCSID and ELIG (or MCSID and RESP).

The variable **ELIG** contains information on what role the carer of the Cohort Member was selected for (eligibility). It is derived based on the information provided at the Household Grid and the main carers of the Cohort Member (usually mother and step/natural father) are prioritised for the role of the Main and Partner. The exact relationship of the Main/Partner to the Cohort Member can be found in the Household Grid in the variable **CREL**.

By merging datasets using ELIG, the corresponding parental interviews will be merged: Main with Main and Partner with Partner interview (where available). However, the actual person responding as Main (or Partner where available) may not be the same with the one of the other sweep. The mcs*_parent_derived dataset provides information on whether there has been a change in the identity (PNUM) of the person responding as Main or Partner.

The RESP variable provides information on whether the individuals selected for the Main/Partner role (ELIG) proceeded with the interview. It can be used to remove missing information across the variables since some individuals did not participate in the interview even if they were eligible (ELIG).

Finally, if the project requires information about both parents (wherever there is a Partner respondent), a solution to increase the sample size of the two parent/carers families is to use the _proxy_partner_ dataset in addition to the parent.

1.6.2 Merging between sweeps: specific person continuity across sweeps (PNUM)

If it is important for the research project that the information comes from the same respondent, then the use of MCSID and PNUM is recommended. The Person Number (PNUM) is the same for an individual across sweeps. The resulting dataset will contain the respondents who have remained the same across different sweeps.

However, respondents who have not been present in later sweeps will not appear in the dataset (_parent_ or _parent_cm_ levels). This will result in a dataset that has lower sample size than the dataset merged using MSCID and ELIG (or MCSID and RESP). Even if the person has moved out, s/he still has a PNUM and is part of the _hhgrid_ dataset (across sweeps).

When merging focusing on specific person across sweeps (PNUM) the _proxy_partner_ datasets will help increase the sample.

2 Datasets of MCS & how to merge them within sweep

This section applies to the long (stacked) format of the datasets of MCS as this is the format that MCS is and will be provided in future sweeps. The examples of code of this section can be applied on any sweep of MCS that is available in a long format. The description of the datasets and the data handling (e.g. merging) of this section have been conducted in R, SPSS Syntax and STATA. The syntax for each of these statistical packages is provided at the end of this guide.

Every **Example code** __ of this section corresponds to respective syntax for R, SPSS Syntax and STATA.

It is important that the reader tries out and syntax provided while following the text.

2.1 Household grid dataset: mcs*_hhgrid

The household grid contains information about the individuals that live or have lived in the same household with the Cohort Member(s). There is one row per person (PNUM) and one row per Cohort Member (CNUM). Here, the data user can find gender and age of each person that lives in the household including for the Cohort Members.

This file contains all the key identifiers that are used in the rest of the datasets of MCS: MCSID, PNUM, CNUM, ELIG, RESP.

MCSID	*PNUM	*CNUM	*ELIG	*RESP
Family identifier	Person number	CM number	Eligibility for interview	Response to Interview
Family 1	1	-1	Main Interview	Main Interview
Family 1	2	-1	Partner Interview	Partner Interview
Family 1	-1	1	-1	-1
Family 2	1	-1	Main Interview	Main Interview
Family 2	2	-1	Partner Interview	Partner Interview
Family 2	3	-1	-1	-1
Family 2	-1	1	-1	-1
Family 2	-1	2	-1	-1

Figure 4: Household Grid identifiers

When looking at the household grid (especially from Sweep 2 onwards) it is important to remember that not all the individuals that have a PNUM still live in the

household. Some people are not part of the household in a subsequent sweep and this is captured by PRES (presence of the individual during the interview).

An important variable is CREL that captures the relationship to the CM.

Many derived variables on relationships within the household are based on CREL. So, if the research project requires information about relationships in the household in general (mcs*_family_derived) or of the main respondents (HTYP) it is worthwhile looking at the datasets that contain the derived variables and the derived variables user guide.

Example code A shows how the data of some families look like.

The Cohort Members do not have PNUM but only CNUM. We can see in PRES that the family structure has changed a lot for some families (where many people have left or passed away) whereas it has remained the same for some others.

2.1.1 Household relationships grid

The relationship of each person to any other person of the household is contained in the relationships grid. This information is very useful as one can draw varying family structures across sweeps, however, it may contain error. The information about the relationship of person X to person Y can contain measurement error firstly due to sensitivity of the question and secondly due to misunderstanding. Namely, a question about whether the relationship of person X to person Y is 'natural sibling' may be sensitive under certain circumstances. Moreover, the terms 'natural sibling' and 'half sibling' may not sound that different even if they imply one parent different between the siblings.

Family	Person Number	Cohort Member Number	Info about person (age, gender, present in hh)	Relation-ship to cohort member	Relationship to Person 1	Relationship to Person 2	Relationship to Person 3	Relationship to Person 4
Family 1	1		…	Parent				
Family 1	2		…	Parent				
Family 1	3		…	Sibling				
Family 1	4		…	…				
Family 1		1	…					
Family 2	1		…	Parent				
Family 2	2		…	Grandparent				
Family 2		1	…					
Family 2		2	…					
Family 3	1		…	Partner				
Family 3		1	…					

Figure 5: Relationships grid

2.1.2 Generating person / cohort members identifiers

For various data manipulations the data user may need a person identifier instead of a family identifier that is the MCSID. It is possible to concatenate (join together) the MCSID with the PNUM or ELIG or CNUM to create a unique person identifier.

Example code B generates a person identifier for each person in the household and a person identifier for each individual of the household including Cohort Members. The syntax concatenates (joins) to create a PID (person ID) for people in the household only and for everyone in the household (including Cohort Members).

The person identifier can be constructed in any dataset, however, the household grid contains all the families regardless if they completed the entire interview of that sweep (full interview or partial completion).

2.2 Cohort member datasets: mcs*_cm_

MCSID	CNUM
Family identifier	Cohort Member Number
Family 1	1
Family 1	2
Family 2	1
Family 3	1

Be aware of more than one Cohort Member per family (i.e. twins & triplets) like Family 1

Figure 6: Cohort member structure dataset

The datasets that only contain _cm_ in the dataset name contain 1 row per cohort member. These can be interview information (mcs*_cm_interview), results of the cognitive assessments (mcs*_cm_cognitive_assessment),

2.2.1 The number of Cohort Members within a sweep (CNUM versus NOCM)

MCSID	VAR
Family identifier	Variable
Family 1	1
Family 2	1
Family 3	2

Figure 7: Family structure dataset

The design of NOCM helps take a closer look to the long format of the MCS dataset. This variable tells us the total number of Cohort Members in each household. Some families have 1 Cohort Member but some have 2 and 3 (twins and triplets).

By running **Example code C**, we can get frequencies of NOCM from the mcs*_family_derived and frequencies of CNUM from the mcs*_cm_derived datasets.

NOCM contains the total numbers of Cohort Members in each household.

CNUM	NOCM = 1	NOCM = 2	NOCM = 3	Total in CNUM
CNUM = 1				
CNUM = 2				
CNUM = 3				

Figure 8: Correspondence of NOCM and CNUM (please use example code C to fill in)

The total number of Cohort Members per household (NOCM) indicates the variation of CNUM. For example, families that have 1 cohort member (NOCM = 1) cannot have CNUM higher than 1.

However, it is possible for a Cohort Member to decide not to participate and in this case the last column of the table would be slightly different, for instance, the CNUM = 1 in a NOCM = 2 family could be missing due to non-response.

2.2.2 Selecting families based on number of Cohort Members

This distinction between CNUM and NOCM can be helpful, depending on the need of the research project, to select households based on the one or the other. For example, the user can use NOCM and keep only families with 1 CM (instead of families with twins or triplets). In this case, the CNUM will be 1 across the file and there will be 11576 families in the dataset (that satisfy NOCM=1). Otherwise, the reseacher may decide to keep only families that have 2 or 3 CMs (NOCM=2 or 3). The CNUM will range from 1 to 3 and it will contain only families with more than one CMs (no singleton families).

If the user selects Cohort Members using CNUM, and selects for example, only CNUM=1 (the first CM of each family), then the file will contain the first Cohort Member of families that have twins or triplets. If the user selects Cohort Members

that have CNUM=2 or CNUM=3 (the second and third CM of each family), then the file will not include any Cohort members that have CNUM=1 including those that belong to families with twins or triplets.

2.3 Parent and proxy partner datasets: mcs*_parent_interview (1 row per parent/carer)

The datasets that contain _parent_ only in their title, refer to information coming from sections where the parent(s) provide information about themselves. The mcs*_parent_interview dataset includes the household questionnaire with information about the household, e.g. language used at home.

MCSID	PNUM	ELIG	RESP
Family identifier	Person number	Eligibility for role of Main/Partner	Response to Interview
Family 1	1	Main Interview	Main Interview
Family 1	6	Partner Interview	Partner Interview
Family 2	1	Main Interview	Main Interview
Family 2	3	Partner Interview	Partner Interview
Family 3	2	Main Interview	Main Interview

Figure 9: Parent structure dataset

2.3.1 mcs*_parent_derived dataset & PNUM vs ELIG/RESP

The mcs*_parent_derived file contains information about the main and partner respondent. The output of **Example code D** focuses on specific families have been selected to illustrate the difference between PNUM, ELIG and RESP. Most of these families have Main and Partner respondents, hence, there are 2 rows per family. In the same output, a Partner respondent is eligible but not available for interview, therefore, the Main answers some questions about him/her (proxy_partner_interview dataset). In another family, Main and Partner respondents are eligible but only the

Main has participated in the interview (RESP=4 'No interview', for the Partner). In the last family, only a Main respondent is eligible for the interview.

2.3.2 Merging mcs*_parent_interview between different sweeps with ELIG or PNUM

If the research project requires data on the _parent_ level from two plus sweeps, the merging of the datasets needs to take into account the MCSID and an identifier of the parent respondent. **Example code E** that demonstrates the merge of the _parent_ datasets of two sweeps uses firstly the MCSID and ELIG and secondly the MCSID and PNUM. This is a good exercise of merging MCS data from different sweeps focusing either on person continuity (PNUM) or information availability (ELIG/RESP).

In **Example code E**, **MCSID** and **ELIG** have been used to match between different sweeps of _parent_ level datasets of MCS. Assuming that the focus is on getting data available about carers of the CM regardless if they are different individuals compared to the previous sweep then MCSID and ELIG can be used. A certain number of Main and Partner respondents gets matched between MCS5 and MCS6 but not all cases. There are two reasons for unmatched cases: non-response on the household level (the family has not participated in the one sweep or the other) or non-response on the person level (for example, the Partner respondent may have moved out, thus there is no Partner respondent but only Main for that family).

Comparing the PNUM of Sweep 5 and Sweep 6, it is possible to see that PNUM has remained the same in more than ___ cases *(please run example code for exact number)*. This means that in these rows the respondent (Main or Partner) is the same person in both sweeps.

In **Example code E**, **MCSID** and **PNUM** have been used assuming that the focus is on the same person answering questions about the family and the parental role.

In this case, more than ___ respondents *(please run example code for exact number)* have remained the same between MCS5 and MCS6 and they have been successfully matched. Both reasons for unmatched cases that occur when merging

with MCSID and ELIG apply here too: some families have not participated but even if they did a second carer of that family may have not participated.

A problem that may arise when merging by PNUM is the fact that a person participated in both sweeps as a parent respondent, however, the role may be different. The role that the person followed during the interview (ELIG) may change from the one sweep to the other. We see that the majority of the respondents out of the total number that participated in both sweeps have the same role (ELIG: Main or Partner). Most of the questions in the parent interview are addressed to both parents. However, some questions are addressed only to the Main respondent (for example, the Household Questionnaire, the Strengths and Difficulties Questionnaire) or only to the Partner. So, even though the same person (parent/carer identified by PNUM) participated in both sweeps, data may be missing (-1 Not applicable) if the individual has participated with a different interview role (ELIG) in each sweep when a particular variable has a Main/Partner only routing.

2.3.3 Composite score per family in a _parent_ level dataset

The research project may require one piece of information about the family regardless how many parents are in the household. At this point it is important to prioritise the derived variables datasets of any structure (_cm_, _parent_, _family_) because they include key information about the family, the survey and the individuals that participated in that sweep.

In **Example code F**, a variable is calculated that contains the mean of the parents'/carers' self-reported health from the variable GENA of MCS6. In the same example code, it is shown how to calculate a variable that contains the highest NVQ of the parents'/carers' using the DNVQ variable from the _parent_derived dataset.

2.3.4 Parent structure dataset: mcs*_proxy_partner_interview

The proxy partner interview can be used to increase the number of Partner respondents in the datasets. The proxy partner interview occurs when the second carer (Partner) of the Cohort Member may not be available to participate in the

interview. This means that the person lives in the household but s/he has not been available for the interview. In these cases, the Main respondent is asked whether s/he is willing to provide information about his/her Partner. The Proxy Partner part of the questionnaire focuses on the key questions that are asked in the parent interview and are important to collect about the person.

The proxy partner module is particularly useful for research focusing on 2-parent or 2-carers families or research that requires information about the same person (PNUM) across time. Through this module information (for example, health, income, employment) is available about the parent who is not able to participate.

Therefore, by merging the information of the _proxy_partner_interview and the _parent_interview it is possible to increase the number of Partner respondents that we have information about as well as the families with 2-parents or 2-carers.

Using the syntax of **Example code G**, we take a look at CREL variable. We see that the majority of the parents that are not available for the interview (but still live in the household) are natural parents. The variables PXRE and PXIN are very helpful in understanding the reasons behind the need for a proxy interview (most common: the partner is working away) and whether the main respondent agreed to provide information in the proxy interview about the partner.

2.3.5 Using data of the mcs*_proxy_partner_interview to maximise the sample size of 2-carers families

Let us suppose that the focus is on increasing the parent sample size with information on general health. For this reason, the variable FXPXGE00 is used from the mcs*_proxy_partner_interview and the variable FPGENA00 from the mcs*_parent_interview. When combining the mcs*_parent_interview with the information coming from the mcs*_proxy_partner_interview, it is not possible to merge using parent identifiers. The two datasets do not contain the same respondents. The mcs*_parent_interview includes the parents that have participated in the interview themselves, whereas, the mcs*_proxy_partner_interview contains

information about the parents that have not been available to participate in the interview. Because of this, we add rows to the dataset (append) rather than merge.

It is important to check what the variable looks like before attempting to combine the datasets. In this case the variables have the same values. The crosstabulation of the variable ELIG with the variable FPGENA00 shows that the dataset contains information from Main and Partner respondents as well as partners that were interviewed through Proxy (Main respondent).

The syntax of **Example code H** highlights the possibilities of handling mcs*_proxy_partner_interview. It is necessary for the same question to exist in both questionnaires: the parent one and the proxy partner.

The CAPI name is likely to be slightly different, so it is good to focus on the questionnaire to identify a pair of questions that tackle the same issue in the parent and the proxy partner interviews. Once the variables have been identified, then appending the one dataset (mcs*_proxy_partner_interview dataset) to the other (mcs*_parent_interview) there will be approximately 200-400 additional respondents in the dataset (Proxy Partners).

2.4 CM structure datasets: mcs*_cm_interview

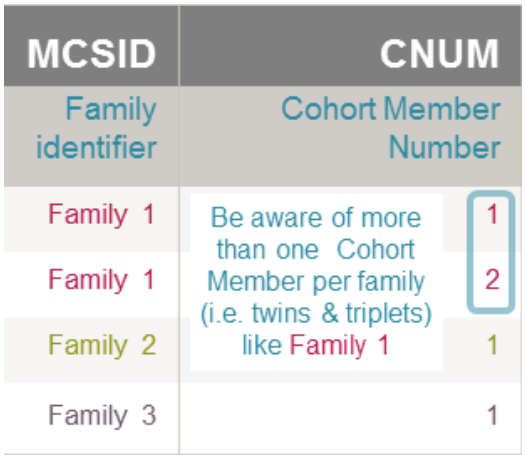

Figure 10: CM structure dataset

The datasets with _cm_ structure contain information that is collected directly from the Cohort Members like the Young Person interview, the Physical Measurements and the Cognitive Assessments.

As there is one row per child, the dataset is in a long (stacked) format. There is a family identifier (MCSID) and Cohort Member identifier (CNUM).

As in the mcs*_cm_derived, CNUM is the identifier for the Cohort Member. The variables CSEX, CDBM, CDBY and CAGE come from the household grid (mcs*_hhgrid) where the data user can find information about the rest of the members of the household. CAGE has been calculated based on the interview date.

2.4.1 Merging mcs*_cm_interview between different sweeps

If the research project requires data on the _cm_ level from two sweeps, the merging needs to take into account the MCSID and the CNUM which is the key identifier for the Cohort Members. The syntax of **Example code I** merges _cm_ level datasets from two different sweeps in a similar way to the _parent_ datasets.

2.5 Parent / CM structure dataset: mcs*_parent_cm_interview

The mcs*_parent_cm_interview is an interesting dataset that contains information that the parent(s) provided about each Cohort Member.

MCSID	PNUM	ELIG	RESP	CNUM
Family identifier	Person number	Eligibility for role of Main/Partner	Response to Interview	Cohort Member Number
Family 1	1	Main Interview	Main Interview	1
Family 1	6	Partner Interview	Partner Interview	1
Family 1	1	Main Interview	Main Interview	2
Family 1	6	Partner Interview	Partner Interview	2
Family 2	1	Main Interview	Main Interview	1
Family 2	3	Partner Interview	Partner Interview	1

Be aware of more than one Cohort Member per family (i.e. twins & triplets) like Family 1

Figure 11: parent_cm structure dataset

The routing of the questions in the parent interview varies based on

a) about whom the question is asked
b) who is asked (both parents, just the Main or just the Partner).

The questionnaire is the most important source of information on whether a variable has _parent_cm_ structure. As a means of illustration of point a), we can take a look at two questions: WALI and BFEV from the first sweep of MCS. The first question asks the parent how satisfied s/he is with life (WALI). The second one asks the parent whether s/he breastfed the Cohort Member (BFEV). The latter question is repeated for each Cohort Member of the family, namely, for the second and third child of families with twins and triplets. So, even if both questions appear in the parent questionnaire, the one will be located in the _parent_dataset (WALI), whereas information about the Cohort Member(s) will be in the _parent_cm_ dataset (BFEV).

Examples of point b) are two questions where the one focuses on the income of both parents and the other one is addressed only towards the one parent. Both of these questions will produce data on the parent level.

Another example is the Strengths and Difficulties Questionnaire (SDQ) that the Main respondent (only) has filled in for each of the Cohort Members of the household. If there is a Partner respondent for a family there will be -1 'Not applicable' for the SDQ for his/her row.

The output of specific variables of a dataset of the _parent_cm_ structure (**Example code J**) illustrates this structure. Firstly, we notice that there are some variables have '-1 Not applicable' in rows of Partner interview. This is because the question has been asked only from the Main respondent. Each family has provided different data depending on the number of parents participating and number of Cohort Members. For example, a family has Main and Partner respondent answering questions about Cohort Member 1. Another family has only Main respondent answer questions about Cohort Member 1. Also, there is a family that has Main and Partner respondents answering questions about 3 Cohort Members.

2.5.1 Merging mcs*_parent_cm_interview between different sweeps

The mcs*_parent_cm_interview contains ELIG/RESP and PNUM as parent identifiers as the mcs*_parent_interview dataset.

Therefore, it can get merged with either MCSID, ELIG, CNUM or MCSID, PNUM, CNUM. A selection of ELIG shows a focus on the data available, whereas selecting PNUM secures person continuity.

Example code K merges the data of mcs*_parent_cm_interview of two difference sweeps with the use of two keys: MCSID and a row ID that is either PNUM & CNUM or ELIG & CNUM. As it happens with the merge of mcs*_parent_interview between sweeps, if we merge by ELIG, different people may have taken the role of the Main or Partner respondent of the CM compared to the previous sweep (specific person gets identified by PNUM). Whereas if we merge by PNUM, the same person may have participated with different roles: namely, the Main in the one sweep and the Partner in another.

2.5.2 Extracting information from a mcs*_parent_cm_interview dataset

There are many pathways for treating data of a mcs*_parent_cm_interview dataset and they depend on what the research aims are. In this section, we provide some ideas and the respective code.

In questions that have been asked only from the Main respondent (like the CSEN and the SDPF of the Strengths and Difficulties Questionnaire) the data user can select only the data of the Main respondent.

If the focus is on using the information provided by both respondents, Main and Partner, then it is possible to use the data as they are or create a composite score. **Example code L** gives an example where a mean of the parent-perceived likelihood to attend University for each CM is calculated. This can be used for example if we would like to examine whether there is difference in how each cohort member has been described by the carers (Main/Partner). In families where there are two parent respondents (Main and Partner) we can use the fact that the information is provided by both carers about the cohort member. The distance ((dis)agreement) between the responses of the two parents or the mean score of the two responses can be used to enhance analysis.

2.5.3 The mcs_longitudinal_family_file: adding outcomes & weights for analysis

The mcs_longitudinal_family_file holds information about every family that has been issued to participate in the MCS. It is therefore a point of reference about the total number of families.

Moreover, it includes families that have left the study in subsequent sweeps due to various reasons (refusal, untraced). This file includes important information about the outcome of the family in each sweep and the weights that can be used for analysis.

Example code M merges a dataset of each structure of MCS6 to the mcs_longitudinal_family_file. This code can be used with any sweep.

3. Examples of data restructures

This chapter provides details and syntax on matching the different datasets of the same sweep while keeping the highest amount of information possible.

The research question dictates the data handling needed. The research scenarios of this chapter are hypothetical and they have been designed to help to illustrate how to restructure the data into one dataset. They are not example of best practice or recommendations for research.

The research scenarios are:

- **3.1** when two datasets with 1-level get matched (example: mcs*_cm_interview & mcs*_parent_interview)
- **3.2** when two datasets of 1-level and 2-level get merged into a 1-level dataset (example: mcs*_parent_cm_interview & mcs*_cm_interview into a _cm_ level dataset)
- **3.3** when two datasets of 1-level and 2-level get merged into a 2-level dataset (example: mcs*_parent_cm_interview & mcs*_cm_interview into a _parent_cm_ level dataset)

In order to get the most out of this section, it is important to have read and tried the preceding chapters.

3.1 Merging two 1-level datasets with different identifiers

As we saw earlier, in MCS, the _parent_ level dataset and the _cm_ level dataset have 1-level. Namely, either 1 row per parent (identifiers PNUM / ELIG & RESP) or 1 row per cohort member (identifier CNUM). It may be required for a research project to merge these two datasets.

The most important decision is which one of the two datasets contains the main variable of interest or the outcome/dependent variable. This way it is possible to select the dataset that will remain the same (dependent / outcome variable) and which dataset will get restructured (independent / predictor variable).

Diagram of merging _cm_ and _parent_ to a _cm_ level dataset

The diagram shows the process of merging two datasets that have 1-level but contain different key identifiers. We need to manipulate the dataset that does not hold the main variable of interest (dependent / outcome variable). Namely, we focus

on restructuring the dataset that has the independent (or outcome) variables. We can either reshape it into wide or calculate a composite variable per family. After the manipulation of the one dataset we simply merge.

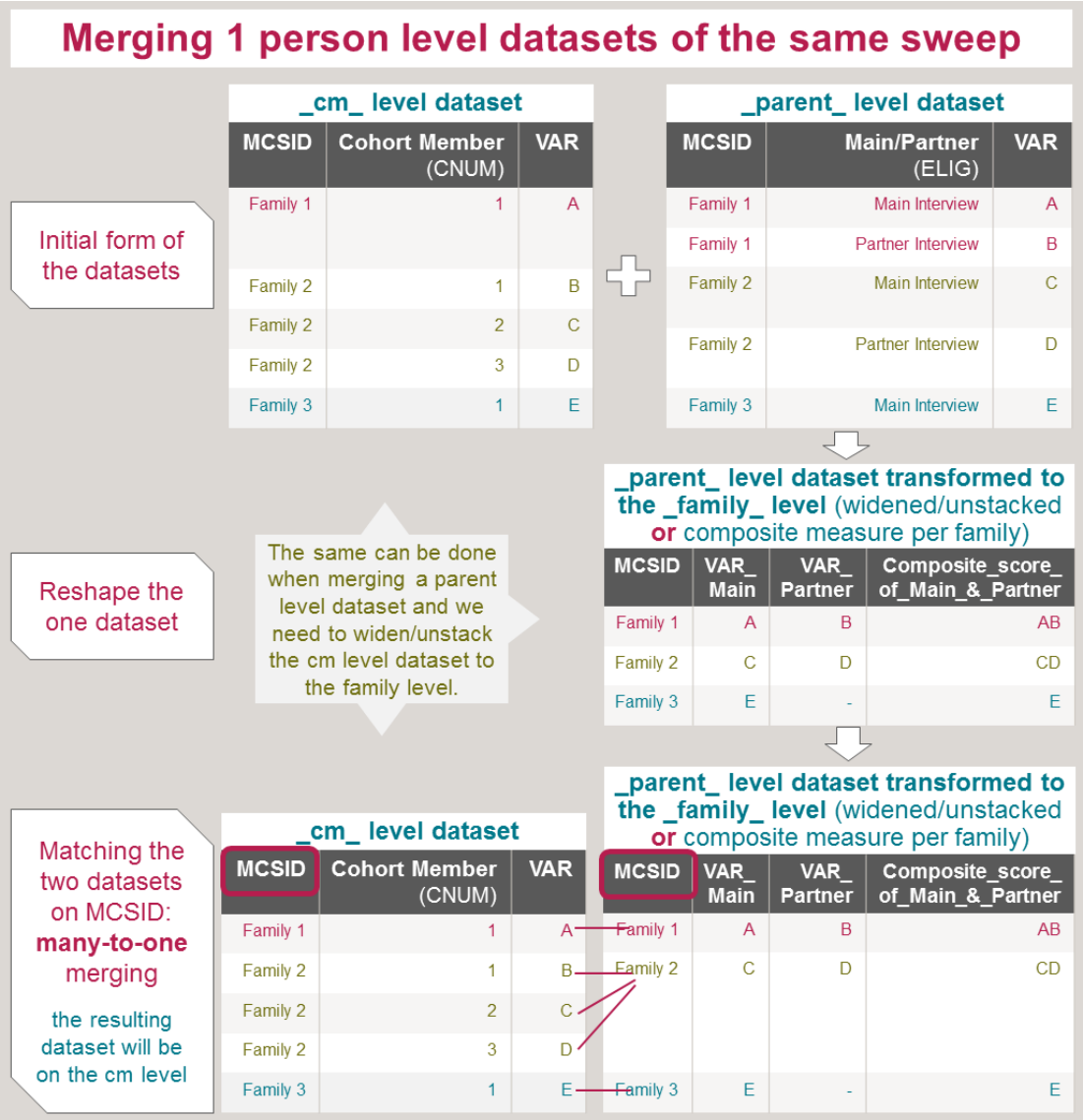

Figure 12: Merging 1 person level datasets of the same sweep (parent with CM)

Merging: _cm_ or _parent <= _cm_ + _parent_

In this example, we want to examine to what extent the parents/carers' general health correlates with the general health reported by the CM.

The variables that we will be using for this example are:

Dataset	Variable name	Variable label

mcs6_cm_interview.sav	FCCGHE00	CM's general level of health
mcs6_parent_interview.sav	FPGENA00	Respondent's general health
mcs6_proxy_partner_interview.sav	FXPXGE00	Describe partner's health

We need the information on general health of both parents, so we will run **code H** that collapses the information coming from the _parents_ dataset as well as from the _proxy_partner_ dataset for the partners that were not present during the interview. After this we follow the diagram. We can either restructure the new _parent_ dataset into wide format or create a composite score of the two parents/carers. **Example code N** does both. It generates a composite score (mean health of parents per household) and then restructures the data into a wide format.

It is important to remember that the variable you select to restructure from long to wide will split the variables in the wide format. In this case we restructure from long to wide using ELIG. The **Example code N** makes sure that ELIG has only 2 levels (Main and Partner&Proxy) because a larger number of levels in ELIG would mean additional variables in the wide format.

As we see in the example that uses the _parent_ dataset, turning into the wide format using the ELIG variable has created 2 variables per variable of the long format. If we turned into wide format the _cm_ dataset (one row per family) using the CNUM, that would create 3 variables per variable of the long format because the CNUM has 3 possible values occurring.

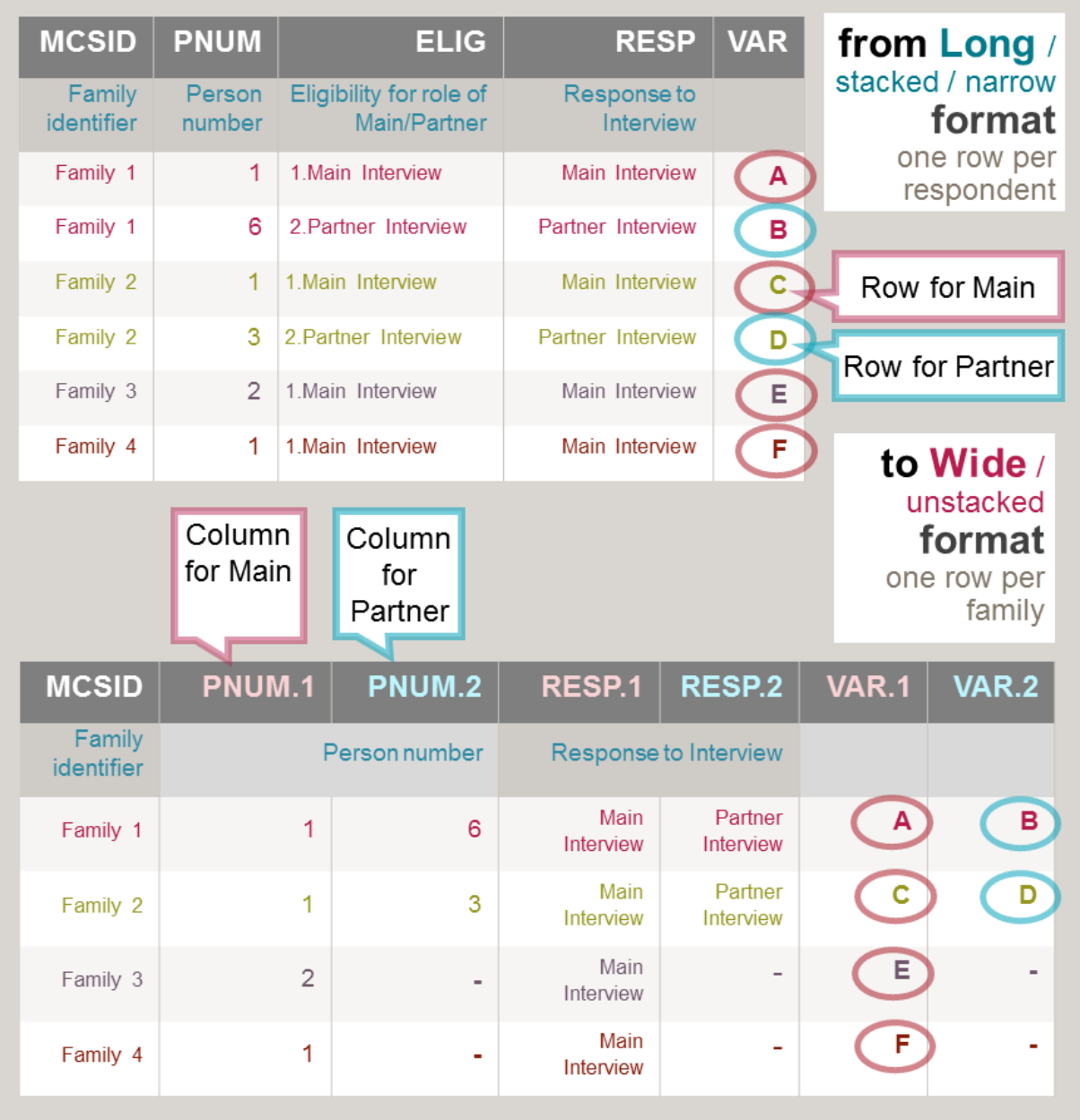

Figure 13: Restructuring long to wide

The **Example code N** concludes with merging the wide format dataset (one family per row) with the _cm_ level dataset. We have one row per CM and in each row there is information with the mean score of the parents of that family.

3.2 Merging 1-level dataset with a 2-level dataset resulting in a 1-level dataset

We may wish to connect information from a 2-level dataset (_parent_cm_) to a 1-level dataset (_cm_ or _parent). As in the previous example we need to select what is our main variable of interest (dependent variable) and restructure the other datasets (that have the independent variables) to match the dataset that our dependent variable is located.

In this section, we assume that the dependent variable is located in the 1-level dataset, so our focus is on ending up with a 1-level dataset (either _parent_ or _cm_ level).

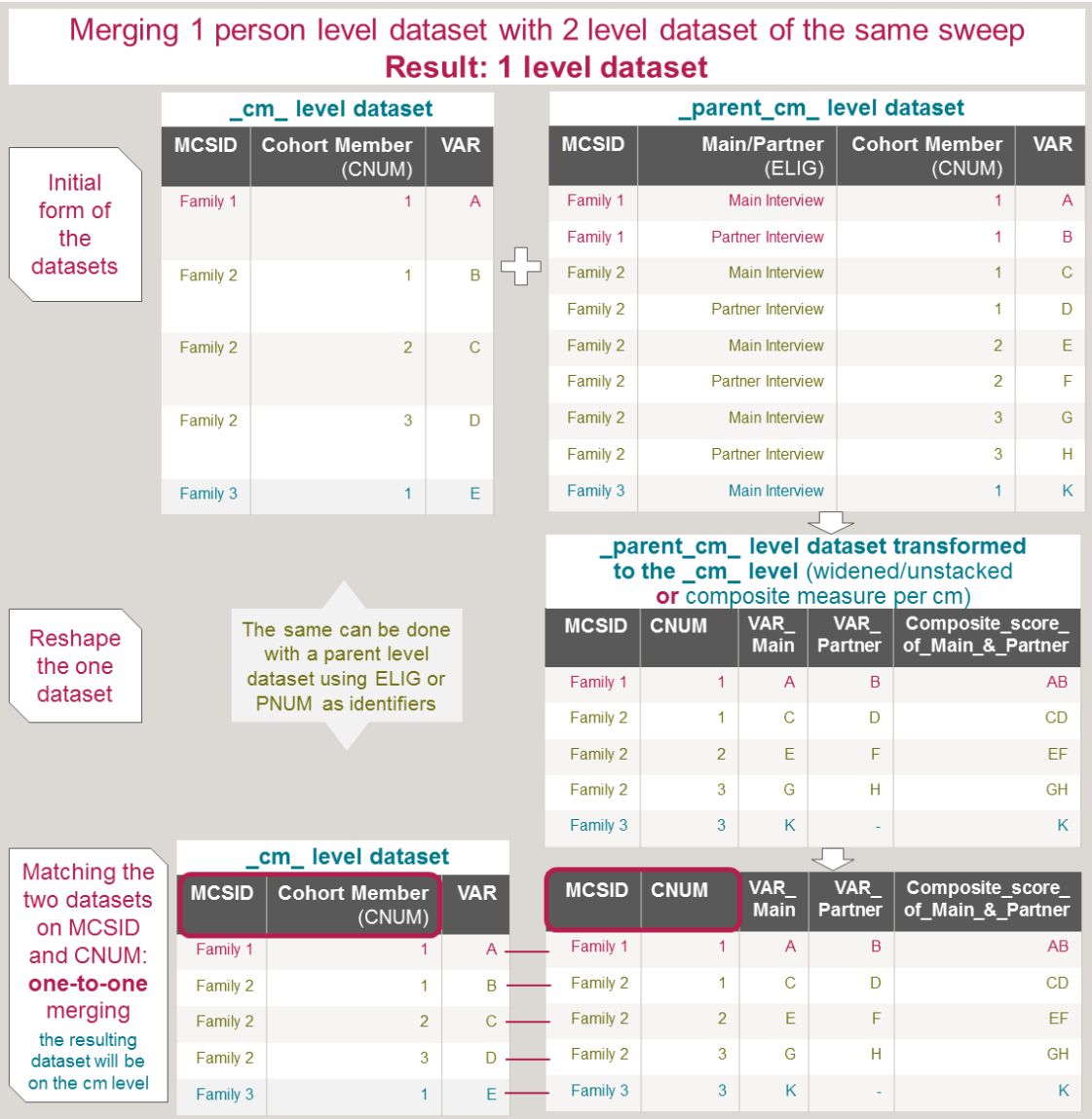

Figure 14: Merging 1 level with a 2 level dataset resulting in a 1 level dataset

The diagram shows a possible path for this. Assuming that our main variable of interest (dependent) is located on the _cm_ dataset, we manipulate the data of our independent variable that is located in the _parent_cm_ dataset. We just need to either restructure the _parent_cm_ or generate a composite score per child or per family, that will get merged to the _cm_ dataset.

Merging: _cm_ <= _cm_ + _parent_cm_

In this example, we want to examine to what extent the parents/carers' aspiration that the CM will go to the University correlates with CM's perception on how likely it is that s/he will go to the University.

The variables that we will be using for this example are:

Dataset	Variable name	Variable label
mcs6_cm_interview.sav	FCSTYU00	How likely is it CM will go to university? (Scale 0-100%)
mcs6_parent_cm_interview.sav	FPASLU00	How (un)likely do you think it is that CM will attend university?

The first step is similar to **Example code L** that generates a composite score for variable ASLU for each child at the _parent_cm_ dataset.This composite score is the mean score of the Main and the Partner for each CM. **Example code O** includes the code of this section.

We create a composite measure of ASLU (mean of parents' perception on how likely it is that the CM will go to the University). We turn the dataset into wide using ELIG, so the information of the parents will be split into two variables: ASLU.1 for the Main and ASLU.2 for the Partner whereas the composite score remains one variable as it is the same for Main and Partner.

You can recalculate the composite score now instead earlier. In the dataset there is one row per child. So, now that the _parent_cm_ has been turned into a _cm_ level dataset, we just merge it with the _cm_interview.

3.3 Merging 1-level dataset with a 2-level dataset resulting in a 2-level dataset

This example is similar to the previous one, however, instead of trying to reduce the _parent_cm_ dataset to a 1-level, we merge it with a 1-level dataset. In case the main variable of interest (dependent variable) is in a 2-level dataset (namely in the _parent_cm_ dataset), then we need to keep the structure of the dataset the same

and merge to another datast, a 1-level dataset (a _parent_ or a _cm_ structure dataset) that has the independent variables.

Diagram of merging _cm_ and _parent_cm_ to a _parent_cm_ level dataset

As we see in the diagram, this merging is the easiest as no dataset requires restructuring.

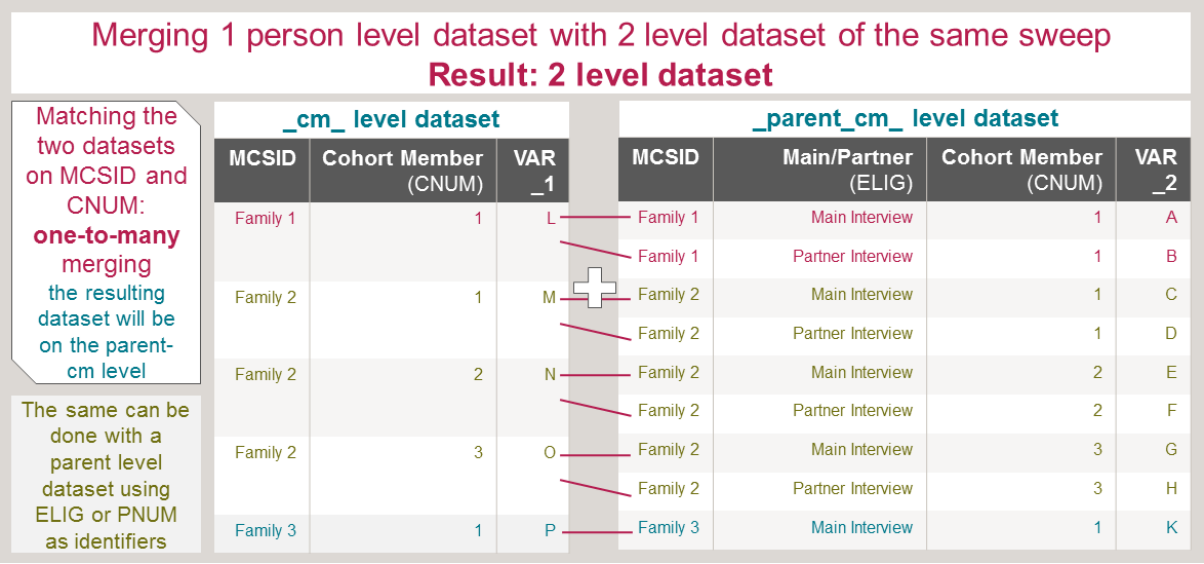

Figure 15: Merging 1 level with a 2 level dataset

Merging: _parent_cm_ <= _cm_ + _parent_cm_

In this example, we want to examine whether there is a correlation between how much the Cohort Member trusts others (_cm_interview) and how close the parents feel that they are to the CM (parent_cm_interview).

The variables that we will be using for this example are:

Dataset	Variable name	Variable label

mcs6_cm_interview.sav	FCSTYU00	How likely is it CM will go to university? (Scale 0-100%)
mcs6_parent_cm_interview.sav	FPSCHC00	Overall, how close would you say you are to CM?

Example code P includes the syntax for this section.

4. Example code with R, SPSS Syntax and STATA

The syntax provided in this section is indicative of different working paths with the datasets of MCS and may contain errors. Users need to build syntax for their own project.

The code provided uses primarily MCS5 and MCS6, however, any long format dataset of MCS can be used. The MCSIDs selected to visualise the data structure were selected for demonstration purposes.

Minor differences between the code provided in SPSS, R and STATA exist due to the different commands and functions in each piece of software.

The calculation of the mean (**Example codes**) in each piece of software may require adjustment to the needs to a particular project as sometimes mean is calculated only for families that there are two carers respondents (Main and Partner) whereas other times for all families (including Main only).

Depending on the research project and type of analysis, clustering of standard errors may be needed if there are multiple rows per family (see User Guide S1-5, sections 2.9 and 6.4.3).

Overview of the example codes

Table 1: Example code contents

Example code	Focus	Question
Example code A	Overview of the hhgrid	How does the household grid (hhgrid)dataset look like?
Example code B	Concatenating MCSID & person identifier to get a unique person identifier	How do I create a unique person identifier by concatenating MCSID & person identifier?
Example code C	Overview of the _family_derived	How does the _family_derived dataset look like?
Example code D	Overview of _parent_derived	How does the _parent_derived dataset look like?
Example code E	Merge _parent_ structure datasets from different sweeps	How do I merge _parent_ structure datasets from different sweeps?
Example code F	Create a composite variable per family in the _parent_ structure file	How do I create a composite variable per family in the _parent_ structure file?
Example code G	Overview of _proxy_partner_interview	How does the _proxy_partner_interview dataset look like?
Example code H	Combining proxy_partner_interview with parent_interview (append)	How do I combine the proxy_partner_interview dataset with the parent_interview dataset?
Example code I	Merge _cm_ structure datasets from different sweeps	How do I merge _cm_ structure datasets from different sweeps?
Example code J	Overview of parent_cm_interview	What does the parent_cm_interview dataset look like?
Example code K	Merge _parent_cm_ level datasets between sweeps	How do I merge the _parent_cm_ level datasets between sweeps?
Example code L	Create a composite variable per child in parent_cm dataset	How do I create a composite variable per child in parent_cm dataset?
Example code M	Merging datasets of different structures to the mcs_longitudinal_family_file	How do I merge datasets of different structures to the mcs_longitudinal_family_file?

Example code N	Merging two 1-level datasets that have different identifiers	Merging two 1-level datasets that have different identifiers (_parent_interview dataset with _cm_interview dataset)
Example code O	Merging a 1-level dataset (_cm_) with a 2-level dataset (_parent_cm_) resulting into a 1-level structure (_cm_)	Merging a 1-level dataset (_cm_) with a 2-level dataset (_parent_cm_) resulting into a 1-level structure (_cm_)
Example code P	Merging a 2-level dataset (_parent_cm_) with a 1-level dataset (_cm_) resulting into a 2-level dataset (_cm_)	Merging a 2-level dataset (_parent_cm_) with a 1-level dataset (_cm_) resulting into a 2-level dataset (_cm_)

R syntax

Setting up folders in R

```
# in case you need to clean the workspace

#rm(list=ls())


# the aim is to use *core* R functions in this syntax

# various packages exist for data management that users may
prefer


# download packages needed

install.packages("foreign")


# load packages needed
```

```
library(foreign)


data_folder_path = "//" # my folder path

# the data of mcs5 and mcs6 need to be in folders mcs6_ and
mcs5_
```

Example code A

```
# --------------------.

# Overview of the hhgrid .

# --------------------------.

mcs6_hhgrid <- read.spss(file= paste(data_folder_path,
"/mcs6_/mcs6_hhgrid.sav", sep = ''), to.data.frame = TRUE,
use.missings=TRUE)

mcs6_hhgrid$MCSID <- trimws(mcs6_hhgrid$MCSID, which =
c('right')) # remove white space from MCSID

str(mcs6_hhgrid$MCSID)

# Overview of PRES / CREL / multiple Cohort Members per
family.

print(mcs6_hhgrid[which(mcs6_hhgrid$MCSID == 'M10002P' |
mcs6_hhgrid$MCSID == 'M10611J' | mcs6_hhgrid$MCSID ==
'M10611J' | mcs6_hhgrid$MCSID == 'M10106W' | mcs6_hhgrid$MCSID
== 'M10063C'), c('MCSID', 'FPNUM00', 'FELIG00', 'FRESP00',
'FCNUM00', 'FHCREL00', 'FHPRES00')])

# NA are the -1 not applicable of the original dataset
```

Example code B

```
# --------------------.

# Concatenating MCSID & person identifier to get a unique
person identifier .

# ------------------------------------------------------------
-------------------------------------------.

mcs6_hhgrid <- read.spss(file= paste(data_folder_path,
"/mcs6_/mcs6_hhgrid.sav", sep = ''), to.data.frame = TRUE)

mcs6_hhgrid$MCSID <- trimws(mcs6_hhgrid$MCSID, which =
c('right')) # remove white space from MCSID

str(mcs6_hhgrid$MCSID)

str(mcs6_hhgrid$FPNUM00)

str(mcs6_hhgrid$FCNUM00)



# a Person ID for each adult in the household (excluding
cohort members) .

mcs6_hhgrid$PnumID <- ifelse(is.na(mcs6_hhgrid$FCNUM00),

                             paste(mcs6_hhgrid$MCSID,
mcs6_hhgrid$FPNUM00, sep = '_P'),

                             mcs6_hhgrid$PnumID <- NA)

head(mcs6_hhgrid$PnumID)
```

```
# a Person ID for each individual of the household (Cohort
Member or other person)

mcs6_hhgrid$PID <- ifelse(is.na(mcs6_hhgrid$FPNUM00),

                          paste(mcs6_hhgrid$MCSID,
mcs6_hhgrid$FCNUM00, sep = '_C'),

                          paste(mcs6_hhgrid$MCSID,
mcs6_hhgrid$FPNUM00, sep = '_P'))

head(mcs6_hhgrid$PID)
```

Example code C

```
# --------------------.

# Overview of the _family_derived .

mcs6_family_derived <- read.spss(file= paste(data_folder_path,
"/mcs6_/mcs6_family_derived.sav", sep = ''), to.data.frame =
TRUE)

table(mcs6_family_derived$FDNOCM00)

table(mcs6_family_derived$FDRSPO00)


# Overview of the _cm_derived .

mcs6_cm_derived <- read.spss(file= paste(data_folder_path,
"/mcs6_/mcs6_cm_derived.sav", sep = ''), to.data.frame = TRUE)

table(mcs6_cm_derived$FCNUM00)
```

Example code D

```
# ------------------.

# Overview of _parent_derived .

# ----------------------------------------.

mcs6_parent_derived <- read.spss(file= paste(data_folder_path,
"/mcs6_/mcs6_parent_derived.sav", sep = ''), to.data.frame =
TRUE)

mcs6_parent_derived$MCSID <- trimws(mcs6_parent_derived$MCSID,
which = c('right'))

mcs6_parent_derived$FELIG00 <-
trimws(mcs6_parent_derived$FELIG00, which = c('right'))

mcs6_parent_derived$FRESP00 <-
trimws(mcs6_parent_derived$FRESP00, which = c('right'))

print(mcs6_parent_derived[which(mcs6_parent_derived$MCSID ==
'M10002P' | mcs6_parent_derived$MCSID == 'M10041W' |
mcs6_parent_derived$MCSID == 'M23136V' |
mcs6_parent_derived$MCSID == 'M10106W' |
mcs6_parent_derived$MCSID == 'M10063C'),

                  c('MCSID', 'FPNUM00', 'FELIG00',
'FRESP00')])
```

Example code E

```
# ------------------.
```

```
# merge _parent_ structure datasets from different sweeps.

# ------------------------------------------------------------
----------------------.


#  ~ ~ ~ ~ ~ ~ merge the two datasets on MCSID & ELIG  ~ ~ ~ ~
~ ~ .

# load the mcs6_parent dataset.

mcs6_parent_interview <- read.spss(file=
paste(data_folder_path, "/mcs6_/mcs6_parent_interview.sav",
sep = ''), to.data.frame = TRUE)

mcs6_parent_interview$MCSID <-
trimws(mcs6_parent_interview$MCSID, which = c('right'))

mcs6_parent_interview$FELIG00 <-
trimws(mcs6_parent_interview$FELIG00, which = c('right'))

# prepare the mcs5_parent_dataset.

mcs5_parent_interview <- read.spss(file=
paste(data_folder_path, "/mcs5_/mcs5_parent_interview.sav",
sep = ''), to.data.frame = TRUE)

mcs5_parent_interview$MCSID <-
trimws(mcs5_parent_interview$MCSID, which = c('right'))

mcs5_parent_interview$EELIG00 <-
trimws(mcs5_parent_interview$EELIG00, which = c('right'))


# create variables to be able to inspect the merge.

mcs6_parent_interview$source_mcs6 <- 'MCS6'
```

```
mcs5_parent_interview$source_mcs5 <- 'MCS5'

mcs6_parent_interview$ELIG <- mcs6_parent_interview$FELIG00

mcs5_parent_interview$ELIG <- mcs5_parent_interview$EELIG00

# merge .

str(mcs6_parent_interview)

str(mcs5_parent_interview)

mcs5_mcs6_parent_interview <- merge(x=mcs6_parent_interview,
y=mcs5_parent_interview,

                                    by.x = c("MCSID", "ELIG"),

                                    by.y = c("MCSID", "ELIG"),

                                    all.x = TRUE, all.y =
TRUE)

# inspect the merge

str(mcs5_mcs6_parent_interview)

mcs5_mcs6_parent_interview$rowsource <-
apply(mcs5_mcs6_parent_interview[c('source_mcs5',
'source_mcs6')], # object to work with

                                                1, # 1 for rows
- 2 for columns

                                                function(x)
paste(na.omit(x), collapse = " ")) # function

table(mcs5_mcs6_parent_interview$rowsource)

# Outcome perusal: Main and Partner respondents (ELIG) in both
sweeps.
```

```
str(mcs5_mcs6_parent_interview$FELIG00)

str(mcs5_mcs6_parent_interview$EELIG00)

mcs5_mcs6_parent_interview$FELIG00 <-
as.factor(mcs5_mcs6_parent_interview$FELIG00)

mcs5_mcs6_parent_interview$EELIG00 <-
as.factor(mcs5_mcs6_parent_interview$EELIG00)

table(mcs5_mcs6_parent_interview$FELIG00,
mcs5_mcs6_parent_interview$EELIG00)


# Let us see how many of the merged Main & Partner respondents
(ELIG) have

# the same PNUM, therefore they are they same person .

mcs5_mcs6_parent_interview$same_respondent <-
ifelse(mcs5_mcs6_parent_interview$FPNUM00 ==
mcs5_mcs6_parent_interview$EPNUM00, 1, 0)

# 1 the same respondent - 0 different respondent

table(mcs5_mcs6_parent_interview$same_respondent,
mcs5_mcs6_parent_interview$ELIG)



#  ~ ~ ~ ~ ~ ~ merge the two datasets on MCSID & PNUM ~ ~ ~ ~
~ ~ .

# load the mcs6_parent dataset.
```

```
mcs6_parent_interview <- read.spss(file=
paste(data_folder_path, "/mcs6_/mcs6_parent_interview.sav",
sep = ''), to.data.frame = TRUE)

mcs6_parent_interview$MCSID <-
trimws(mcs6_parent_interview$MCSID, which = c('right'))

mcs6_parent_interview$FELIG00 <-
trimws(mcs6_parent_interview$FELIG00, which = c('right'))

# prepare the mcs5_parent_dataset.

mcs5_parent_interview <- read.spss(file=
paste(data_folder_path, "/mcs5_/mcs5_parent_interview.sav",
sep = ''), to.data.frame = TRUE)

mcs5_parent_interview$MCSID <-
trimws(mcs5_parent_interview$MCSID, which = c('right'))

mcs5_parent_interview$EELIG00 <-
trimws(mcs5_parent_interview$EELIG00, which = c('right'))



# create variables to be able to inspect the merge.

mcs6_parent_interview$source_mcs6 <- 'MCS6'

mcs5_parent_interview$source_mcs5 <- 'MCS5'

mcs6_parent_interview$PNUM <- mcs6_parent_interview$FPNUM00

mcs5_parent_interview$PNUM <- mcs5_parent_interview$EPNUM00

# merge .

str(mcs6_parent_interview)

str(mcs5_parent_interview)
```

```
mcs5_mcs6_parent_interview <- merge(x=mcs6_parent_interview,
y=mcs5_parent_interview,
                                    by.x = c("MCSID", "PNUM"),
                                    by.y = c("MCSID", "PNUM"),
                                    all.x = TRUE, all.y =
TRUE)

# inspect the merge

str(mcs5_mcs6_parent_interview)

mcs5_mcs6_parent_interview$rowsource <-
apply(mcs5_mcs6_parent_interview[c('source_mcs5',
'source_mcs6')], # object to work with
                                                1, # 1 for rows
- 2 for columns
                                                function(x)
paste(na.omit(x), collapse = "&")) # function

table(mcs5_mcs6_parent_interview$rowsource)

# Outcome perusal: parents/carers (PNUM) in both sweeps.


# Let us see how many of the merged parents/carers respondents
(PNUM) have

# the same role in the interview (ELIG: Main or Partner) .

table(mcs5_mcs6_parent_interview$EELIG00,
mcs5_mcs6_parent_interview$FELIG00, useNA = 'ifany')

# respondents that were not eligible in the one or the other
sweep have NA
```

```
# the crosstabulation shows the respondents whether
respondents have the

# same or different eligibility between sweeps
```

Example code F

```
# -------------------.

# Create a composite variable per family in the _parent_
structure file .

# ------------------------------------------------------------
-----------------------------------.

mcs6_parent_interview <- read.spss(file=
paste(data_folder_path, "/mcs6_/mcs6_parent_interview.sav",
sep = ''), to.data.frame = TRUE)

mcs6_parent_interview$MCSID <-
trimws(mcs6_parent_interview$MCSID, which = c('both'));
str(mcs6_parent_interview$MCSID)


# this syntax creates a mean of Main and Partner responses on
GENA variable .

# You can use other functions instead of MEAN, like SD, MIN,
MAX, etc.

str(mcs6_parent_interview$FPGENA00)

table(mcs6_parent_interview$FPGENA00, useNA = 'ifany')

mcs6_parent_interview$GENA <-
as.numeric(mcs6_parent_interview$FPGENA00)
```

```
table(mcs6_parent_interview$GENA, useNA = 'ifany')

mcs6_parent_interview_small <-
mcs6_parent_interview[c("MCSID", "FPNUM00", "FELIG00",
"FPGENA00")]

mcs6_parent_interview_small$GENA_num <-
as.numeric(mcs6_parent_interview_small$FPGENA00)

str(mcs6_parent_interview_small)

mcs6_parent_interview_small_composite <-
aggregate(mcs6_parent_interview_small$GENA_num ~
mcs6_parent_interview_small$MCSID, FUN=mean, na.rm=TRUE,
na.action="na.omit")

str(mcs6_parent_interview_small_composite)

colnames(mcs6_parent_interview_small_composite) <- c('MCSID',
'GENA_composite')

mcs6_parent_interview_small_composite$MCSID <-
as.character(trimws(mcs6_parent_interview_small_composite$MCSI
D, which = c('both')))

# connect to the rest of the data

mcs6_parent_interview_with_GENA <- merge(x =
mcs6_parent_interview_small_composite,

                                         y =
mcs6_parent_interview_small,

                                         by.x = 'MCSID', by.y =
'MCSID', all = TRUE)

str(mcs6_parent_interview_with_GENA)

# let's take a look at the outcome
```

```
print(mcs6_parent_interview_with_GENA[which(mcs6_parent_interv
iew_with_GENA$MCSID == 'M10002P' |
mcs6_parent_interview_with_GENA$MCSID == 'M10611J' |
mcs6_parent_interview_with_GENA$MCSID == 'M10106W' |
mcs6_parent_interview_with_GENA$MCSID == 'M10063C'),

                        c('MCSID', 'FPNUM00', 'FELIG00',
'FPGENA00', 'GENA_num', 'GENA_composite')])



# this syntax selects the higher NVQ of Main and Partner
respondents.

# You can use other functions instead of MEAN, like SD, MIN,
MAX, etc.

mcs6_parent_derived <- read.spss(file= paste(data_folder_path,
"/mcs6_/mcs6_parent_derived.sav", sep = ''), to.data.frame =
TRUE)

mcs6_parent_derived$MCSID <- trimws(mcs6_parent_derived$MCSID,
which = c('both')); str(mcs6_parent_derived$MCSID)

table(mcs6_parent_derived$FDNVQ00, useNA='ifany')

mcs6_parent_derived$NVQ_num <-
as.numeric(mcs6_parent_derived$FDNVQ00);
table(mcs6_parent_derived$NVQ_num, useNA='ifany')

mcs6_parent_derived$NVQ_num[mcs6_parent_derived$NVQ_num >= 6]
<- NA

table(mcs6_parent_derived$NVQ_num, useNA='ifany')

mcs6_parent_derived_composite <-
aggregate(mcs6_parent_derived$NVQ_num ~
mcs6_parent_derived$MCSID, FUN=max, na.rm=TRUE,
na.action="na.omit")
```

```
str(mcs6_parent_derived_composite)

colnames(mcs6_parent_derived_composite) <- c('MCSID',
'NVQ_composite')

mcs6_parent_derived_composite$MCSID <-
as.character(trimws(mcs6_parent_derived_composite$MCSID, which
= c('both')))

# connect to the rest of the data

mcs6_parent_derived_with_NVQ <- merge(x =
mcs6_parent_derived_composite,

                                      y =
mcs6_parent_derived,

                                      by.x = 'MCSID', by.y
= 'MCSID', all = TRUE)

str(mcs6_parent_derived_with_NVQ)

# let's take a look at the outcome

print(mcs6_parent_derived_with_NVQ[which(mcs6_parent_derived_w
ith_NVQ$MCSID == 'M10002P' |
mcs6_parent_derived_with_NVQ$MCSID == 'M10611J' |
mcs6_parent_derived_with_NVQ$MCSID == 'M10106W' |
mcs6_parent_derived_with_NVQ$MCSID == 'M10063C'),

                                    c('MCSID', 'FPNUM00',
'FELIG00', 'FDNVQ00', 'NVQ_num', 'NVQ_composite')])
```

Example code G

```
# -------------------.
```

```
# Overview of _proxy_partner_interview .

# ------------------------------------------------------.

mcs6_proxy_partner_interview <- read.spss(file=
paste(data_folder_path,
"/mcs6_/mcs6_proxy_partner_interview.sav", sep = ''),
to.data.frame = TRUE)

mcs6_proxy_partner_interview$MCSID <-
trimws(mcs6_proxy_partner_interview$MCSID, which = c('both'));
str(mcs6_proxy_partner_interview$MCSID)


table(mcs6_proxy_partner_interview$FXCREL00)

table(mcs6_proxy_partner_interview$FXCREL00,
mcs6_proxy_partner_interview$FXPSEX00)

table(mcs6_proxy_partner_interview$FXPXRE00)

table(mcs6_proxy_partner_interview$FXPXIN00)
```

Example code H

```
# --------------------.


# Combining proxy_partner_interview with parent_interview .

# ------------------------------------------------------------
----------------------.
```

```
# we keep only rows where the Main agreed to provide
information about the non-available Partner.

nrow(mcs6_proxy_partner_interview) # ___ observations

mcs6_proxy_partner_interview_subset <-
subset(mcs6_proxy_partner_interview,


mcs6_proxy_partner_interview$FXPXIN00 == "Continue with PROXY
interview  ")

nrow(mcs6_proxy_partner_interview_subset)

table(mcs6_proxy_partner_interview_subset$FXPXGE00)

# we rename the variable to the variable name that is used in
the parent_interview dataset.

names(mcs6_proxy_partner_interview_subset)[names(mcs6_proxy_pa
rtner_interview_subset) == "FXPXGE00"] <- "FPGENA00"

mcs6_proxy_partner_interview_for_connection_to_parent <-
mcs6_proxy_partner_interview_subset[c("MCSID", "FPNUM00",
"FELIG00", "FPGENA00")]


# open the parent_interview dataset and keep only the
variables needed .

mcs6_parent_interview <- read.spss(file=
paste(data_folder_path, "/mcs6_/mcs6_parent_interview.sav",
sep = ''), to.data.frame = TRUE)

mcs6_parent_interview$MCSID <-
trimws(mcs6_parent_interview$MCSID, which = c('both'));
str(mcs6_parent_interview$MCSID)
```

```
mcs6_parent_interview_small <-
mcs6_parent_interview[c("MCSID", "FPNUM00", "FELIG00",
"FPGENA00")]


# we add cases/rows to the dataset - append.

str(mcs6_parent_interview_small)

str(mcs6_proxy_partner_interview_for_connection_to_parent)

table(mcs6_parent_interview_small$FPGENA00)

table(mcs6_proxy_partner_interview_for_connection_to_parent$FP
GENA00)

mcs6_parent_plus_proxy_interview <-
rbind(mcs6_parent_interview_small,
mcs6_proxy_partner_interview_for_connection_to_parent)

str(mcs6_parent_plus_proxy_interview)

table(mcs6_parent_plus_proxy_interview$FPGENA00)

# there are 2 factor level 'excellent' because the value label
wording is slightly different

# they can get collapsed together

table(mcs6_parent_plus_proxy_interview$FELIG00)
```

Example code I

```
# --------------------.

# merge _cm_ structure datasets from different sweeps.
```

```
# ------------------------------------------------------------
---------------.

# prepare the mcs6_cm dataset.

mcs6_cm_interview <- read.spss(file= paste(data_folder_path,
"/mcs6_/mcs6_cm_interview.sav", sep = ''), to.data.frame =
TRUE)

mcs6_cm_interview$MCSID <- trimws(mcs6_cm_interview$MCSID,
which = c('both')); str(mcs6_cm_interview$MCSID)

mcs6_cm_interview$CNUM <- mcs6_cm_interview$FCNUM00;
table(mcs6_cm_interview$CNUM)


# prepare the mcs5_cm_dataset.

mcs5_cm_interview <- read.spss(file= paste(data_folder_path,
"/mcs5_/mcs5_cm_interview.sav", sep = ''), to.data.frame =
TRUE)

mcs5_cm_interview$MCSID <- trimws(mcs5_cm_interview$MCSID,
which = c('both')); str(mcs6_cm_interview$MCSID)

mcs5_cm_interview$CNUM <- mcs5_cm_interview$ECNUM00;
table(mcs5_cm_interview$CNUM)


# merge the two datasets on MCSID & CNUM .

mcs6_cm_interview$sweep_6 <- 'Sweep 6'

mcs5_cm_interview$sweep_5 <- 'Sweep 5'

mcs5_mcs6_cm_interview <- merge(x = mcs6_cm_interview,

                                y = mcs5_cm_interview,
```

```
                                    by.x = c('MCSID', 'CNUM'),

                                    by.y = c('MCSID', 'CNUM'), all
= TRUE)

# Outcome perusal: cohort members in both datasets.

table(mcs5_mcs6_cm_interview$sweep_6,
mcs5_mcs6_cm_interview$sweep_5, useNA='ifany')

table(mcs5_mcs6_cm_interview$CNUM)

table(mcs5_mcs6_cm_interview$FCNUM00,
mcs5_mcs6_cm_interview$ECNUM00, useNA='ifany')
```

Example code J

```
# --------------------.

# Overview of _parent_cm_interview .

# ------------------------------------------------.

# Parent's interview about the CM(s) of the household.

mcs6_parent_cm_interview <- read.spss(file=
paste(data_folder_path, "/mcs6_/mcs6_parent_cm_interview.sav",
sep = ''), to.data.frame = TRUE)

mcs6_parent_cm_interview$MCSID <-
trimws(mcs6_parent_cm_interview$MCSID, which = c('both'));
str(mcs6_parent_cm_interview$MCSID)

table(mcs6_parent_cm_interview$FCNUM00)
```

```
# Overview of specific cases on key variables

print(mcs6_parent_cm_interview[which(mcs6_parent_cm_interview$
MCSID == 'M10002P' |


mcs6_parent_cm_interview$MCSID == 'M10611J' |


mcs6_parent_cm_interview$MCSID == 'M10106W' |


mcs6_parent_cm_interview$MCSID == 'M10063C'),

                                    c('MCSID', 'FPNUM00',
'FELIG00', 'FCNUM00', 'FCCSEX00', 'FCCAGE00', 'FPSDPF00',
'FPASLU00')])
```

Example code K

```
# --------------------.

# Merge _parent_cm_ level datasets between sweeps.

# ------------------------------------------------------------
---------------.

# Create row identifier to connect for _parent_cm_ datasets.

# ROWid = (ELIG or PNUM) + (CNUM Child 1/2/3) .


# ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~  ELIG + CNUM = ROWid ~ ~ ~ ~ ~ ~ ~ ~
~ ~ ~ ~ .

# prepare mcs6_parent_cm_interview .
```

```
mcs6_parent_cm_interview <- read.spss(file=
paste(data_folder_path, "/mcs6_/mcs6_parent_cm_interview.sav",
sep = ''), to.data.frame = TRUE)

mcs6_parent_cm_interview$MCSID <-
trimws(mcs6_parent_cm_interview$MCSID, which = c('both'));
str(mcs6_parent_cm_interview$MCSID)

table(mcs6_parent_cm_interview$FCNUM00)

table(mcs6_parent_cm_interview$FPNUM00)



mcs6_parent_cm_interview$ROWid <-
ifelse(mcs6_parent_cm_interview$FELIG00 == "Main Interview ",

                                        paste('M_C',
mcs6_parent_cm_interview$FCNUM00, sep = ''),

                                        paste('P_C',
mcs6_parent_cm_interview$FCNUM00, sep = '')

)

table(mcs6_parent_cm_interview$ROWid, useNA='ifany')

# Check that the ROWid matches the crosstabulation of ELIG &
CNUM .

table(mcs6_parent_cm_interview$FCNUM00,
mcs6_parent_cm_interview$FELIG00)



# prepare mcs5_parent_cm_interview .

mcs5_parent_cm_interview <- read.spss(file=
paste(data_folder_path, "/mcs5_/mcs5_parent_cm_interview.sav",
sep = ''), to.data.frame = TRUE)
```

```
mcs5_parent_cm_interview$MCSID <-
trimws(mcs5_parent_cm_interview$MCSID, which = c('both'));
str(mcs5_parent_cm_interview$MCSID)

table(mcs5_parent_cm_interview$ECNUM00)

table(mcs5_parent_cm_interview$EPNUM00)



mcs5_parent_cm_interview$ROWid <-
ifelse(mcs5_parent_cm_interview$EELIG00 == "Main Interview ",

                                        paste('M_C',
mcs5_parent_cm_interview$ECNUM00, sep = ''),

                                        paste('P_C',
mcs5_parent_cm_interview$ECNUM00, sep = '')

)

table(mcs5_parent_cm_interview$ROWid, useNA='ifany')

# Check that the ROWid matches the crosstabulation of ELIG &
CNUM .

table(mcs5_parent_cm_interview$ECNUM00,
mcs5_parent_cm_interview$EELIG00)



# Merge .

mcs6_parent_cm_interview$sweep_6 <- 'Sweep 6'

mcs5_parent_cm_interview$sweep_5 <- 'Sweep 5'

mcs5_mcs6_parent_cm_interview <- merge(x =
mcs6_parent_cm_interview,

                                y = mcs5_parent_cm_interview,
```

```
                                by.x = c('MCSID', 'ROWid'),

                                by.y = c('MCSID', 'ROWid'),
all = TRUE)

table(mcs5_mcs6_parent_cm_interview$sweep_6,
mcs5_mcs6_parent_cm_interview$sweep_5, useNA='ifany')

# Outcome perusal: parents (Main/Partner providing information
about

# each of the cohort members) are in both sweeps.




# ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~  PNUM + CNUM = ROWid ~ ~ ~ ~ ~ ~ ~ ~ ~
~ ~ ~ ~ .

# prepare mcs6_parent_cm_interview .

mcs6_parent_cm_interview <- read.spss(file=
paste(data_folder_path, "/mcs6_/mcs6_parent_cm_interview.sav",
sep = ''), to.data.frame = TRUE)

mcs6_parent_cm_interview$MCSID <-
trimws(mcs6_parent_cm_interview$MCSID, which = c('both'));
str(mcs6_parent_cm_interview$MCSID)

table(mcs6_parent_cm_interview$FCNUM00)

table(mcs6_parent_cm_interview$FPNUM00)


mcs6_parent_cm_interview$ROWid <- paste('P',
mcs6_parent_cm_interview$FPNUM00, '_C',
mcs6_parent_cm_interview$FCNUM00, sep = '')
```

```
table(mcs6_parent_cm_interview$ROWid, useNA='ifany')

# Check that the ROWid matches the crosstabulation of PNUM &
CNUM .

table(mcs6_parent_cm_interview$FPNUM00,
mcs6_parent_cm_interview$FCNUM00)


# prepare mcs5_parent_cm_interview .

mcs5_parent_cm_interview <- read.spss(file=
paste(data_folder_path, "/mcs5_/mcs5_parent_cm_interview.sav",
sep = ''), to.data.frame = TRUE)

mcs5_parent_cm_interview$MCSID <-
trimws(mcs5_parent_cm_interview$MCSID, which = c('both'));
str(mcs5_parent_cm_interview$MCSID)

table(mcs5_parent_cm_interview$ECNUM00)

table(mcs5_parent_cm_interview$EPNUM00)


mcs5_parent_cm_interview$ROWid <- paste('P',
mcs5_parent_cm_interview$EPNUM00, '_C',
mcs5_parent_cm_interview$ECNUM00, sep = '')

table(mcs5_parent_cm_interview$ROWid, useNA='ifany')

# Check that the ROWid matches the crosstabulation of PNUM &
CNUM .

table(mcs5_parent_cm_interview$EPNUM00,
mcs5_parent_cm_interview$ECNUM00)
```

```
# Merge .

mcs6_parent_cm_interview$sweep_6 <- 'Sweep 6'

mcs5_parent_cm_interview$sweep_5 <- 'Sweep 5'

mcs5_mcs6_parent_cm_interview <- merge(x =
mcs6_parent_cm_interview,

                                       y =
mcs5_parent_cm_interview,

                                       by.x = c('MCSID',
'ROWid'),

                                       by.y = c('MCSID',
'ROWid'), all = TRUE)

table(mcs5_mcs6_parent_cm_interview$sweep_6,
mcs5_mcs6_parent_cm_interview$sweep_5, useNA='ifany')

# Outcome perusal: Individuals (PNUM) provide information

# about the cohort member(s) in both sweeps (either as Main or
Partner respondent).
```

Example code L

```
# --------------------.

# Create a composite variable per child in parent_cm dataset.

# ------------------------------------------------------.

# this syntax creates a mean of Main and Partner responses on
ASLU variable .
```

```
# You can use other functions instead of MEAN, like SD, MIN,
MAX, etc.

mcs6_parent_cm_interview <- read.spss(file=
paste(data_folder_path, "/mcs6_/mcs6_parent_cm_interview.sav",
sep = ''), to.data.frame = TRUE)

mcs6_parent_cm_interview$MCSID <-
trimws(mcs6_parent_cm_interview$MCSID, which = c('both'));
str(mcs6_parent_cm_interview$MCSID)

# Mean score of ASLU assessment by Main/Partner respondent(s)
of the cohort member

table(mcs6_parent_cm_interview$FPASLU00, useNA='ifany')

mcs6_parent_cm_interview$ASLU_num <-
as.numeric(mcs6_parent_cm_interview$FPASLU00)

table(mcs6_parent_cm_interview$ASLU_num, useNA = 'ifany')

mcs6_parent_cm_interview_small <-
mcs6_parent_cm_interview[c('ASLU_num', 'MCSID', 'FCNUM00')]

mcs6_parent_cm_interview_composite <-
aggregate(mcs6_parent_cm_interview_small$ASLU_num ~
mcs6_parent_cm_interview_small$MCSID +
mcs6_parent_cm_interview_small$FCNUM00 , FUN=mean, na.rm=TRUE,
na.action="na.omit")

str(mcs6_parent_cm_interview_composite)

colnames(mcs6_parent_cm_interview_composite) <- c('MCSID',
'FCNUM00', 'ASLU_mean')

mcs6_parent_cm_interview_with_ASLU <- merge(x =
mcs6_parent_cm_interview_composite,
```

```
                                                y =
mcs6_parent_cm_interview,

                                                by.x = c('MCSID',
'FCNUM00' ),

                                                by.y = c('MCSID',
'FCNUM00'), all = TRUE)

# Overview of specific cases on composite score for each
Cohort Member

print(mcs6_parent_cm_interview_with_ASLU[which(mcs6_parent_cm_
interview_with_ASLU$MCSID == 'M10002P' |
mcs6_parent_cm_interview_with_ASLU$MCSID == 'M10611J' |
mcs6_parent_cm_interview_with_ASLU$MCSID == 'M10106W' |
mcs6_parent_cm_interview_with_ASLU$MCSID == 'M10063C'),

                                            c('MCSID', 'FPNUM00',
'FELIG00', 'FCNUM00', 'FPASLU00', 'ASLU_num', 'ASLU_mean')])
```

Example code M

```
# --------------------.


# Merging datasets of different structures to the
mcs_longitudinal_family_file .

# ------------------------------------------------------------
-----------------------------------------------.

mcs_longitudinal_family_file <- read.spss(file=
paste(data_folder_path,
"/mcs6_/mcs_longitudinal_family_file.sav", sep = ''),
to.data.frame = TRUE)
```

```
mcs_longitudinal_family_file$MCSID <-
trimws(mcs_longitudinal_family_file$MCSID, which = c('both'))

mcs_longitudinal_family_file$All_sweeps <- 'longitudinal'

# merge with a _parent_ level dataset .

mcs6_parent_derived <- read.spss(file= paste(data_folder_path,
"/mcs6_/mcs6_parent_derived.sav", sep = ''), to.data.frame =
TRUE)

mcs6_parent_derived$MCSID <- trimws(mcs6_parent_derived$MCSID,
which = c('both'))

mcs6_parent_derived$Sweep_6 <- 'Sweep 6'

mcs6_parent_derived_plus_longitudinal_file <- merge(x =
mcs6_parent_derived,

                                        y =
mcs_longitudinal_family_file,

                                        by.x = c('MCSID'),

                                        by.y = c('MCSID'), all
= TRUE)

table(mcs6_parent_derived_plus_longitudinal_file$Sweep_6,
mcs6_parent_derived_plus_longitudinal_file$All_sweeps,
useNA='ifany')



# merge with a _cm_ level dataset .

mcs6_cm_derived <- read.spss(file= paste(data_folder_path,
"/mcs6_/mcs6_cm_derived.sav", sep = ''), to.data.frame = TRUE)

mcs6_cm_derived$MCSID <- trimws(mcs6_cm_derived$MCSID, which =
c('both'))
```

```
mcs6_cm_derived$Sweep_6 <- 'Sweep 6'

mcs6_cm_derived_plus_longitudinal_file <- merge(x =
mcs6_cm_derived,

                                                y =
mcs_longitudinal_family_file,

                                                by.x =
c('MCSID'),

                                                by.y =
c('MCSID'), all = TRUE)

table(mcs6_cm_derived_plus_longitudinal_file$Sweep_6,
mcs6_cm_derived_plus_longitudinal_file$All_sweeps,
useNA='ifany')



# merge with a _family_ level dataset .

mcs6_family_derived <- read.spss(file= paste(data_folder_path,
"/mcs6_/mcs6_family_derived.sav", sep = ''), to.data.frame =
TRUE)

mcs6_family_derived$MCSID <- trimws(mcs6_family_derived$MCSID,
which = c('both'))

mcs6_family_derived$Sweep_6 <- 'Sweep 6'

mcs6_family_derived_plus_longitudinal_file <- merge(x =
mcs6_family_derived,

                                                y =
mcs_longitudinal_family_file,

                                                by.x =
c('MCSID'),
```

```
                                                    by.y =
c('MCSID'), all = TRUE)

table(mcs6_family_derived_plus_longitudinal_file$Sweep_6,
mcs6_family_derived_plus_longitudinal_file$All_sweeps,
useNA='ifany')



# merge with a _parent_cm_ level dataset .

mcs6_parent_cm_interview <- read.spss(file=
paste(data_folder_path, "/mcs6_/mcs6_parent_cm_interview.sav",
sep = ''), to.data.frame = TRUE)

mcs6_parent_cm_interview$MCSID <-
trimws(mcs6_parent_cm_interview$MCSID, which = c('both'))

mcs6_parent_cm_interview$Sweep_6 <- 'Sweep 6'

mcs6_parent_cm_interview_plus_longitudinal_file <- merge(x =
mcs6_parent_cm_interview,

                                                    y =
mcs_longitudinal_family_file,

                                                    by.x =
c('MCSID'),

                                                    by.y =
c('MCSID'), all = TRUE)

table(mcs6_parent_cm_interview_plus_longitudinal_file$Sweep_6,
mcs6_parent_cm_interview_plus_longitudinal_file$All_sweeps,
useNA='ifany')
```

Example code N

```
# --------------------.

# Example I - merging two 1-level datasets that have different
identifiers.

# Please run example code H to get the _parent_ level dataset
below.

str(mcs6_parent_plus_proxy_interview)

mcs6_parent_plus_proxy_interview$GENA_num <-
as.numeric(mcs6_parent_plus_proxy_interview$FPGENA00)

# create a composite score of the general health of the Main &
Partner.

# Mean self-assessed health of Main/Partner respondent(s) in
each family

mcs6_parent_plus_proxy_interview_small <-
mcs6_parent_plus_proxy_interview[c('MCSID', 'GENA_num')]

mcs6_parent_plus_proxy_interview_composite <-
aggregate(mcs6_parent_plus_proxy_interview$GENA_num ~
mcs6_parent_plus_proxy_interview$MCSID, FUN=mean)

str(mcs6_parent_plus_proxy_interview_composite)

colnames(mcs6_parent_plus_proxy_interview_composite) <-
c('MCSID', 'GENA_composite')

# connect to the rest of the data

mcs6_parent_plus_proxy_interview_with_GENA <- merge(x =
mcs6_parent_plus_proxy_interview,

                                        y =
mcs6_parent_plus_proxy_interview_composite,
```

```
                                        by.x = 'MCSID', by.y
= 'MCSID', all = TRUE)

str(mcs6_parent_plus_proxy_interview_with_GENA)

table(mcs6_parent_plus_proxy_interview_with_GENA$FELIG00,
useNA='ifany')

mcs6_parent_plus_proxy_interview_with_GENA$ELIG[mcs6_parent_pl
us_proxy_interview_with_GENA$FELIG00 %in% c("Main Interview
")] <- 'Main'

mcs6_parent_plus_proxy_interview_with_GENA$ELIG[mcs6_parent_pl
us_proxy_interview_with_GENA$FELIG00 %in% c("Partner Interview
")] <- 'Partner'

mcs6_parent_plus_proxy_interview_with_GENA$ELIG[mcs6_parent_pl
us_proxy_interview_with_GENA$FELIG00 %in% c("Proxy
Interview")] <- 'Proxy'

mcs6_parent_wide <-
reshape(mcs6_parent_plus_proxy_interview_with_GENA,

        timevar = "ELIG",

        idvar = c("MCSID", "GENA_composite"),

        direction = "wide")

names(mcs6_parent_wide)

head(mcs6_parent_wide)


# merge _cm_ level with the wide restructured parent dataset
(one row per family).
```

```
mcs6_cm_interview <- read.spss(file= paste(data_folder_path,
"/mcs6_/mcs6_cm_interview.sav", sep = ''), to.data.frame =
TRUE)

mcs6_cm_interview$MCSID <- trimws(mcs6_cm_interview$MCSID,
which = c('both')); str(mcs6_cm_interview$MCSID)



# merge the two datasets on MCSID & CNUM .

mcs6_cm_interview$Source_cm <- 'CM_long__one_row_per_child'

mcs6_parent_wide$Source_parent <-
'Parent__wide_one_row_per_family'

mcs6_cm_interview_with_parent_GENA <- merge(x =
mcs6_cm_interview,

                                y = mcs6_parent_wide,

                                by.x = 'MCSID',

                                by.y = 'MCSID', all = TRUE)

table(mcs6_cm_interview_with_parent_GENA$Source_cm,
mcs6_cm_interview_with_parent_GENA$Source_parent,
useNA='ifany')

# comparison between parents' general health and CM's.

table(mcs6_cm_interview_with_parent_GENA$GENA_composite,
mcs6_cm_interview_with_parent_GENA$FCCGHE00)
```

Example code O

```
# --------------------.
```

```
# Example II - merging a 1-level dataset (_cm_) with a 2-level
dataset (_parent_cm_) resulting into a 1-level structure
(_cm_).

# Please check example code L to get the _parent_cm_ level
dataset, it is similar.

# we generate a composite measure of ASLU like in the example
code L.

str(mcs6_parent_cm_interview_with_ASLU)

mcs6_parent_cm_interview_with_ASLU <-
mcs6_parent_cm_interview_with_ASLU[c('MCSID', 'FPNUM00',
'FELIG00', 'FCNUM00', 'FPASLU00', 'ASLU_mean')]

# we turn the _parent_cm_ dataset into wide format (one row
per CM).

mcs6_parent_cm_interview_with_ASLU$ELIG[mcs6_parent_cm_intervi
ew_with_ASLU$FELIG00 %in% c("Main Interview ")] <- 'Main'

mcs6_parent_cm_interview_with_ASLU$ELIG[mcs6_parent_cm_intervi
ew_with_ASLU$FELIG00 %in% c("Partner Interview      ")] <-
'Partner'

mcs6_parent_cm_interview_with_ASLU_wide <-
reshape(mcs6_parent_cm_interview_with_ASLU,

                                                    timevar =
'ELIG',

                                                    idvar =
c('MCSID', 'FCNUM00', 'ASLU_mean'),

                                                    direction =
'wide')

names(mcs6_parent_cm_interview_with_ASLU_wide)
```

```
head(mcs6_parent_cm_interview_with_ASLU_wide)


# merge _cm_interview level with the wide restructured parent
dataset (one row per family).

mcs6_cm_interview <- read.spss(file= paste(data_folder_path,
"/mcs6_/mcs6_cm_interview.sav", sep = ''), to.data.frame =
TRUE)

mcs6_cm_interview$MCSID <- trimws(mcs6_cm_interview$MCSID,
which = c('both')); str(mcs6_cm_interview$MCSID)


# merge the two datasets on MCSID & CNUM .

mcs6_cm_interview$Source_cm <- 'CM_long__one_row_per_child'

mcs6_parent_cm_interview_with_ASLU_wide$Source_parent_cm <-
'Parent_cm_wide_one_row_per_child'

mcs6_parent_cm_interview_with_ASLU_wide_with_cm <- merge(x =
mcs6_cm_interview,
                                                y =
mcs6_parent_cm_interview_with_ASLU_wide,
                                                by.x = c('MCSID',
'FCNUM00'),
                                                by.y = c('MCSID',
'FCNUM00'), all = TRUE)

table(mcs6_parent_cm_interview_with_ASLU_wide_with_cm$Source_c
m,
mcs6_parent_cm_interview_with_ASLU_wide_with_cm$Source_parent_
cm, useNA='ifany')
```

```
# comparison between parents' perception of CM's likelihood to
go to the University and CM's perception on the same topic.

table(mcs6_parent_cm_interview_with_ASLU_wide_with_cm$FCSTYU00
, mcs6_parent_cm_interview_with_ASLU_wide_with_cm$ASLU_mean,
useNA='ifany')
```

Example code P

```
# --------------------.

# Example III - merging a 2-level dataset (_parent_cm_) with a
1-level dataset (_cm_) resulting into a 2-level dataset
(_cm_).

mcs6_cm_interview <- read.spss(file= paste(data_folder_path,
"/mcs6_/mcs6_cm_interview.sav", sep = ''), to.data.frame =
TRUE)

mcs6_parent_cm_interview <- read.spss(file=
paste(data_folder_path, "/mcs6_/mcs6_parent_cm_interview.sav",
sep = ''), to.data.frame = TRUE)

mcs6_cm_interview$MCSID <- trimws(mcs6_cm_interview$MCSID,
which = c('both')); str(mcs6_cm_interview$MCSID)

mcs6_parent_cm_interview$MCSID <-
trimws(mcs6_parent_cm_interview$MCSID, which = c('both'));
str(mcs6_parent_cm_interview$MCSID)


# merge the two datasets on MCSID & CNUM .
```

```
mcs6_cm_interview$Source_cm <- 'CM dataset'

mcs6_parent_cm_interview$Source_parent_cm <- 'Parent_CM
dataset'

mcs6_parent_cm_interview_plus_cm <- merge(x =
mcs6_cm_interview,

                                  y = mcs6_parent_cm_interview,

                                  by.x = c('MCSID', 'FCNUM00'),

                                  by.y = c('MCSID', 'FCNUM00'),
all = TRUE)

table(mcs6_parent_cm_interview_plus_cm$Source_cm,

      mcs6_parent_cm_interview_plus_cm$Source_parent_cm, useNA
= 'ifany')

table(mcs6_parent_cm_interview_plus_cm$FCTRST0A,

      mcs6_parent_cm_interview_plus_cm$FPSCHC00,
useNA='ifany')
```

SPSS syntax

* Setting up folders in SPSS .

```
file handle mcs5_folder /name = 'user_folder_path_\_of_mcs5'.

file handle mcs6_folder /name = 'user_folder_path_\_of_mcs6'.

file handle mcs_working_folder /name =
'user_folder_path_\_of_mcs_work_in_progress'.


* =============== .

SET TNUMBERS BOTH.

SET OVARS BOTH.

SET TVARS BOTH.
```

* Example code A

```
* --------------------.

* Overview of the hhgrid .

* --------------------------.

GET FILE = 'mcs6_folder/mcs6_hhgrid.sav'.

* Overview of PRES / CREL / multiple Cohort Members per
family.

TEMPORARY.
```

```
SELECT IF ((MCSID EQ 'M10002P') OR (MCSID EQ 'M10063C') OR
(MCSID EQ 'M10611J') OR (MCSID EQ 'M10106W') OR (MCSID EQ
'M10063C')).

SUMMARIZE

/TABLES = MCSID FPNUM00 FELIG00 FRESP00 FCNUM00 FHCREL00
FHPRES00

/FORMAT = VALIDLIST NOCASENUM TOTAL

/TITLE = 'Overview of specific cases on key variables'

/MISSING = VARIABLE

/CELLS = COUNT.
```

* Example code B

```
* --------------------.

* Concatenating MCSID & person identifier to get a unique
person identifier .

* ------------------------------------------------------------
-------------------------------------------.

GET FILE = 'mcs6_folder/mcs6_hhgrid.sav'.


* a Person ID for each adult in the household (excluding
cohort members) .

STRING PnumID (A9).
```

```
COMPUTE PnumID = concat(rtrim(MCSID), ltrim(string(FPNUM00,
F2))  ).

EXECUTE.

SORT CASES BY PnumID.


* a Person ID for each individual of the household (Cohort
Member or other person) .

STRING CMrow (A3).

IF (FCNUM00 EQ 1) CMrow = '_C1'.

IF (FCNUM00 EQ 2) CMrow = '_C2'.

IF (FCNUM00 EQ 3) CMrow = '_C3'.

STRING PID (A11).

COMPUTE PID = concat(rtrim(MCSID), '_', ltrim(string(FPNUM00,
F2)) ).

EXECUTE.

IF (FCNUM00 EQ 1 OR 2 OR 3) PID = concat(rtrim(MCSID),
ltrim(CMrow)).
```

* Example code C

```
* --------------------.

* Overview of the _family_derived .

GET FILE = 'mcs6_folder/mcs6_family_derived.sav'.
```

```
FREQUENCIES FDNOCM00.

FREQUENCIES FDRSPO00.


* Overview of the _cm_derived .

GET FILE = 'mcs6_folder/mcs6_cm_derived.sav'.

FREQUENCIES FCNUM00 .
```

* Example code D

```
* --------------------.

* Overview of _parent_derived .

* ----------------------------------------.

GET FILE = 'mcs6_folder/mcs6_parent_derived.sav'.


TEMPORARY.

SELECT IF ((MCSID EQ 'M10002P') OR (MCSID EQ 'M10041W') OR
(MCSID EQ 'M23136V') OR (MCSID EQ 'M10106W') OR (MCSID EQ
'M10063C')).

SUMMARIZE

/TABLES = MCSID FPNUM00 FELIG00 FRESP00

/FORMAT = VALIDLIST NOCASENUM TOTAL

/TITLE = 'Overview of specific cases on key variables'

/MISSING = VARIABLE
```

```
/CELLS = COUNT.
```

* Example code E

```
* --------------------.

* merge _parent_ structure datasets from different sweeps.

* ------------------------------------------------------------
----------------------.


*  ~ ~ ~ ~ ~ ~ merge the two datasets on MCSID & ELIG  ~ ~ ~ ~
~ ~ .

* prepare the mcs6_parent dataset.

GET FILE = 'mcs6_folder/mcs6_parent_interview.sav'.

FREQUENCIES FPNUM00 FELIG00.

COMPUTE ELIG = FELIG00.

EXECUTE.

SORT CASES BY MCSID ELIG (A).

SAVE OUTFILE =
'mcs_working_folder/mcs6_parent_interview_cross_sweep_merging.
sav'.

* prepare the mcs5_parent_dataset.

GET FILE = 'mcs5_folder/mcs5_parent_interview.sav'.

FREQUENCIES EPNUM00 EELIG00.
```

```
COMPUTE ELIG = EELIG00.

EXECUTE.

SORT CASES BY MCSID ELIG (A).

SAVE OUTFILE =
'mcs_working_folder/mcs5_parent_interview_cross_sweep_merging.
sav'.

* merge .

GET FILE =
'mcs_working_folder/mcs6_parent_interview_cross_sweep_merging.
sav'.

MATCH FILES /FILE=*

  /IN source_mcs6


/FILE='mcs_working_folder/mcs5_parent_interview_cross_sweep_me
rging.sav'

  /IN source_mcs5

  /BY MCSID ELIG.

EXECUTE.

* Outcome perusal: Main and Partner respondents (ELIG) in both
sweeps.

CROSSTABS source_mcs6 BY source_mcs5 .

FREQUENCIES ELIG.

CROSSTABS FELIG00 BY EELIG00 .

* Let us see how many of the merged Main & Partner respondents
(ELIG) have
```

```
* the same PNUM, therefore they are they same person .

* Values in the diagonal of the crosstabulation are
respondents who are the

* same in both sweeps.

TEMPORARY.

SELECT IF ELIG = 1.

CROSSTABS FPNUM00 BY EPNUM00 .

IF (FPNUM00 EQ EPNUM00) SAME_RESPONDENT = 1.

IF SYSMIS(SAME_RESPONDENT) SAME_RESPONDENT = 0.

VARIABLE LABELS SAME_RESPONDENT 'Is the Main/Partner
respondent the same btw S5 & S6?'.

VALUE LABELS SAME_RESPONDENT 1 'Same' 0 'Different or missing
data'.

FREQUENCIES SAME_RESPONDENT.

CROSSTABS ELIG BY SAME_RESPONDENT.

SAVE OUTFILE =
'mcs_working_folder/mcs5_mcs6_parent_interview_by_ELIG.sav'.


*  ~ ~ ~ ~ ~ ~ merge the two datasets on MCSID & PNUM ~ ~ ~ ~
~ ~ .

* prepare the mcs6_parent_ dataset.

GET FILE = 'mcs6_folder/mcs6_parent_interview.sav'.

COMPUTE PNUM = FPNUM00 .
```

```
SORT CASES BY MCSID PNUM (A).

SAVE OUTFILE =
'mcs_working_folder/mcs6_parent_interview_cross_sweep_merging.
sav'.

* prepare the mcs5_parent_dataset.

GET FILE = 'mcs5_folder/mcs5_parent_interview.sav'.

COMPUTE PNUM = EPNUM00.

EXECUTE.

SORT CASES BY MCSID PNUM (A).

SAVE OUTFILE =
'mcs_working_folder/mcs5_parent_interview_cross_sweep_merging.
sav'.

* merge.

GET FILE =
'mcs_working_folder/mcs6_parent_interview_cross_sweep_merging.
sav'.

MATCH FILES /FILE=*

  /IN source_mcs6


/FILE='mcs_working_folder/mcs5_parent_interview_cross_sweep_me
rging.sav'

  /IN source_mcs5

  /BY MCSID PNUM.

EXECUTE.

* Outcome perusal: parents/carers (PNUM) in both sweeps.
```

```
CROSSTABS source_mcs6 BY source_mcs5 .

* Let us see how many of the merged parents/carers respondents
(PNUM) have

* the same role in the interview (ELIG: Main or Partner) .

IF (FELIG00 EQ EELIG00) SAME_ELIGIBILITY = 1.

IF SYSMIS(SAME_ELIGIBILITY) SAME_ELIGIBILITY = 0.

VARIABLE LABELS SAME_ELIGIBILITY 'Is the role at the interview
(ELIG) the same btw S5 & S6?'.

VALUE LABELS SAME_ELIGIBILITY 1 'Same' 0 'Different or missing
data'.

FREQUENCIES SAME_ELIGIBILITY.

CROSSTABS PNUM BY SAME_ELIGIBILITY.

SAVE OUTFILE =
'mcs_working_folder/mcs5_mcs6_parent_interview_by_PNUM.sav'.
```

* Example code F

```
* --------------------.

* Create a composite variable per family in the _parent_
structure file .

* ------------------------------------------------------------
-----------------------------------.

* this syntax creates a mean of Main and Partner responses on
GENA variable .

* You can use other functions instead of MEAN, like SD, MIN,
MAX, etc.
```

```
GET FILE = 'mcs6_folder/mcs6_parent_interview.sav'.

AGGREGATE

  outfile=*

  overwrite=yes

  mode=addvariables

  /break= MCSID

  /composite_GENA = MEAN(FPGENA00)

  /groupsize = N.

VARIABLE LABELS composite_GENA 'Mean score of GENA of
Main/Partner respondent(s) per family'.

VARIABLE LABELS groupsize 'Number of respondents (Main only,
Partner only, or Main&Partner) providing information in GENA'.

* Let's take a look at the result .

TEMPORARY.

SELECT IF ((MCSID EQ 'M10002P') OR (MCSID EQ 'M10611J') OR
(MCSID EQ 'M10106W') OR (MCSID EQ 'M10063C')).

SUMMARIZE

/TABLES = MCSID

FPNUM00

FELIG00

FRESP00

FPGENA00

composite_GENA
```

```
groupsize

/FORMAT = VALIDLIST NOCASENUM TOTAL

/TITLE = 'Overview of specific cases on composite score for
each family (GENA = general health)'

/MISSING = VARIABLE

/CELLS = COUNT.


* this syntax selects the higher NVQ of Main and Partner
respondents.

* You can use other functions instead of MEAN, like SD, MIN,
MAX, etc.

GET FILE = 'mcs6_folder/mcs6_parent_derived.sav'.

FREQUENCIES FDNVQ00 .

IF (FDNVQ00 GE 1 AND FDNVQ00 LE 5) NVQ = FDNVQ00.

EXECUTE.

CROSSTABS FDNVQ00 BY NVQ.

AGGREGATE

  outfile=*

  overwrite=yes

  mode=addvariables

  /break= MCSID

  /composite_NVQ = MAX(NVQ)

  /groupsize = N.
```

```
VARIABLE LABELS composite_NVQ 'Highest NVQ of Main/Partner
respondent(s) in each family'.

VARIABLE LABELS groupsize 'Number of respondents (Main only,
Partner only, or Main&Partner) with information on NVQ'.

* Let's take a look at the result .

TEMPORARY.

SELECT IF ((MCSID EQ 'M10002P') OR (MCSID EQ 'M10611J') OR
(MCSID EQ 'M10451L') OR (MCSID EQ 'M10106W') OR (MCSID EQ
'M10063C')).

SUMMARIZE

/TABLES = MCSID

FPNUM00

FELIG00

FRESP00

FDNVQ00

composite_NVQ

groupsize

/FORMAT = VALIDLIST NOCASENUM TOTAL

/TITLE = 'Overview of specific cases on composite score for
each family (highest NVQ)'

/MISSING = VARIABLE

/CELLS = COUNT.
```

* Example code G

```
* --------------------.


* Overview of _proxy_partner_interview .

* ------------------------------------------------------.


GET FILE = 'mcs6_folder/mcs6_proxy_partner_interview.sav'.

FREQUENCIES FXCREL00 .

CROSSTABS FXCREL00 BY FXPSEX00 .

FREQUENCIES FXPXRE00 FXPXIN00 .
```

* Example code H

```
* --------------------.


* Combining proxy_partner_interview with parent_interview .

* ------------------------------------------------------------
----------------------.

* we keep only rows where the Main agreed to provide
information about the non-available Partner.

SELECT IF FXPXIN00 EQ 1 .

FREQUENCIES FXPXGE00.

* we rename the variable to the variable name that is used in
the parent_interview dataset.
```

```
RENAME VARIABLES FXPXGE00 = FPGENA00 .

SORT CASES BY MCSID (A).

SAVE OUTFILE =
'mcs_working_folder/mcs6_proxy_partner_interview_for_connectio
n_to_parent.sav'

/KEEP

MCSID

FELIG00

FRESP00

FXCREL00

FPGENA00 .


* open the parent_interview dataset and keep only the
variables needed .

GET FILE = 'mcs6_folder/mcs6_parent_interview.sav'.

FREQUENCIES FPGENA00.

SAVE OUTFILE =
'mcs_working_folder/mcs6_parent_interview_for_connection_to_pr
oxy.sav'

/KEEP

MCSID

FELIG00

FRESP00
```

```
FPCREL00

FPGENA00 .


* open the reduced parent_interview dataset.

GET FILE =
'mcs_working_folder/mcs6_parent_interview_for_connection_to_pr
oxy.sav'.

* we add cases/rows to the dataset.

ADD FILES /FILE=*


/FILE='mcs_working_folder\mcs6_proxy_partner_interview_for_con
nection_to_parent.sav'.

EXECUTE.

FREQUENCIES FPGENA00.

CROSSTABS FPGENA00 BY FELIG00.

SAVE OUTFILE =
'mcs_working_folder/mcs6_parent_plus_proxy_interview.sav'.
```

* Example code I

```
* --------------------.

* merge _cm_ structure datasets from different sweeps.

* ------------------------------------------------------------
---------------.

* prepare the mcs6_cm dataset.
```

```
GET FILE = 'mcs6_folder/mcs6_cm_interview.sav'.

FREQUENCIES FCNUM00 .

COMPUTE CNUM = FCNUM00.

EXECUTE.

SORT CASES BY MCSID CNUM (A).

SAVE OUTFILE =
'mcs_working_folder/mcs6_cm_interview_cross_sweep_merging.sav'
.

* prepare the mcs5_cm_dataset.

GET FILE = 'mcs5_folder/mcs5_cm_interview.sav'.

FREQUENCIES ECNUM00.

COMPUTE CNUM = ECNUM00.

EXECUTE.

SORT CASES BY MCSID CNUM (A).

SAVE OUTFILE =
'mcs_working_folder/mcs5_cm_interview_cross_sweep_merging.sav'
.

* merge the two datasets on MCSID & CNUM .

GET FILE =
'mcs_working_folder/mcs6_cm_interview_cross_sweep_merging.sav'
.

MATCH FILES /FILE=*

  /IN source_mcs6
```

```
/FILE='mcs_working_folder/mcs5_cm_interview_cross_sweep_mergin
g.sav'
  /IN source_mcs5
  /BY MCSID CNUM.
EXECUTE.
* Outcome perusal: cohort members in both datasets.
CROSSTABS source_mcs6 BY source_mcs5 .
FREQUENCIES CNUM.
CROSSTABS FCNUM00 BY ECNUM00 .
SAVE OUTFILE =
'mcs_working_folder/mcs5_mcs6_cm_interview.sav'.
```

* Example code J

```
* --------------------.
* Overview of _parent_cm_interview .
* ------------------------------------------------.
* Parent's interview about the CM(s) of the household.
GET FILE = 'mcs6_folder/mcs6_parent_cm_interview.sav'.
FREQUENCIES FCNUM00 .

TEMPORARY.
```

```
SELECT IF ((MCSID EQ 'M10002P') OR (MCSID EQ 'M10611J') OR
(MCSID EQ 'M10106W') OR (MCSID EQ 'M10063C')).

SUMMARIZE

/TABLES = MCSID

FCNUM00

FELIG00

FCNUM00

FCCSEX00

FCCAGE00

FPSDPF00

FPASLU00

/FORMAT = VALIDLIST NOCASENUM TOTAL

/TITLE = 'Overview of specific cases on key variables'

/MISSING = VARIABLE

/CELLS = COUNT.
```

* Example code K

```
* --------------------.

* Merge _parent_cm_ level datasets between sweeps.

* ------------------------------------------------------------
---------------.

* Create row identifier to connect for _parent_cm_ datasets.
```

```
* ROWid = (ELIG or PNUM) + (CNUM Child 1/2/3) .


* ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~  ELIG + CNUM = ROWid ~ ~ ~ ~ ~ ~ ~ ~
~ ~ ~ ~ .

* prepare mcs6_parent_cm_interview .

GET FILE = 'mcs6_folder/mcs6_parent_cm_interview.sav'.

STRING ELIGrow (A1).

IF (FELIG00 EQ 1) ELIGrow = "M".

IF (FELIG00 EQ 2) ELIGrow = "P".

COMPUTE CNUM = FCNUM00.

EXECUTE.

CROSSTABS CNUM BY ELIGrow.

STRING ROWid (A2).

COMPUTE ROWid = concat (rtrim(ELIGrow), ltrim(string(CNUM,
F1))).

EXECUTE.

* Check that the ROWid matches the crosstabulation of ELIG &
CNUM .

FREQUENCIES ROWid.

SORT CASES BY MCSID ROWid (A).

SAVE OUTFILE =
'mcs_working_folder/mcs6_parent_cm_interview_cross_sweep_mergi
ng.sav' .
```

```
* prepare mcs5_parent_cm_interview .

GET FILE = 'mcs5_folder/mcs5_parent_cm_interview.sav'.

STRING ELIGrow (A1).

IF (EELIG00 EQ 1) ELIGrow = "M".

IF (EELIG00 EQ 2) ELIGrow = "P".

COMPUTE CNUM = ECNUM00.

EXECUTE.

CROSSTABS CNUM BY ELIGrow.

STRING ROWid (A2).

COMPUTE ROWid = concat (rtrim(ELIGrow), ltrim(string(CNUM,
F1))).

EXECUTE.

* Check that the ROWid matches the crosstabulation of ELIG &
CNUM .

FREQUENCIES ROWid.

SORT CASES BY MCSID ROWid (A).

SAVE OUTFILE =
'mcs_working_folder/mcs5_parent_cm_interview_cross_sweep_mergi
ng.sav' .

* Merge .

GET FILE =
'mcs_working_folder/mcs6_parent_cm_interview_cross_sweep_mergi
ng.sav'.

MATCH FILES /FILE=*
```

```
  /IN source_mcs6


/FILE='mcs_working_folder/mcs5_parent_cm_interview_cross_sweep
_merging.sav'

  /IN source_mcs5

  /BY MCSID ROWid.

EXECUTE.

* Outcome perusal: Parents (Main/Partner providing information
about

* each of the cohort members) are in both sweeps.

CROSSTABS source_mcs6 BY source_mcs5 .

SAVE OUTFILE =
'mcs_working_folder/mcs5_mcs6_parent_cm_interview_by_ELIG.sav'
.




* ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~  PNUM + CNUM = ROWid ~ ~ ~ ~ ~ ~ ~ ~
~ ~ ~ ~ .

* prepare mcs6_parent_cm_interview .

GET FILE = 'mcs6_folder/mcs6_parent_cm_interview.sav'.

STRING CMrow (A3).

IF (FCNUM00 EQ 1) CMrow = '_C1'.

IF (FCNUM00 EQ 2) CMrow = '_C2'.

IF (FCNUM00 EQ 3) CMrow = '_C3'.
```

```
EXECUTE.

STRING Prow (A3).

COMPUTE Prow = concat (rtrim('P'), ltrim(string(FPNUM00,
F2))).

EXECUTE.

STRING ROWid(A6).

COMPUTE ROWid = concat (rtrim(Prow), ltrim(CMrow)).

EXECUTE.

* Let's check that the totals match.

CROSSTABS FPNUM00 BY FCNUM00.

FREQUENCIES ROWid.

SORT CASES BY MCSID ROWid (A).

SAVE OUTFILE =
'mcs_working_folder/mcs6_parent_cm_interview_cross_sweep_mergi
ng.sav'.


* prepare mcs5_parent_cm_interview .

GET FILE = 'mcs5_folder/mcs5_parent_cm_interview.sav'.

STRING CMrow (A3).

IF (ECNUM00 EQ 1) CMrow = '_C1'.

IF (ECNUM00 EQ 2) CMrow = '_C2'.

IF (ECNUM00 EQ 3) CMrow = '_C3'.

EXECUTE.
```

```
STRING Prow (A3).

COMPUTE Prow = concat (rtrim('P'), ltrim(string(EPNUM00,
F2))).

EXECUTE.

STRING ROWid(A6).

COMPUTE ROWid = concat (rtrim(Prow), ltrim(CMrow)).

EXECUTE.

* Let's check that the totals match.

CROSSTABS EPNUM00 BY ECNUM00.

FREQUENCIES ROWid.

SORT CASES BY MCSID ROWid (A).

SAVE OUTFILE =
'mcs_working_folder/mcs5_parent_cm_interview_cross_sweep_mergi
ng.sav'.

* Merge .

GET FILE =
'mcs_working_folder/mcs6_parent_cm_interview_cross_sweep_mergi
ng.sav'.

MATCH FILES /FILE=*

  /IN source_mcs6


/FILE='mcs_working_folder/mcs5_parent_cm_interview_cross_sweep
_merging.sav'

  /IN source_mcs5
```

```
  /BY MCSID ROWid.

EXECUTE.

CROSSTABS source_mcs6 BY source_mcs5 .

* Outcome perusal: Only ____ Individuals (PNUM) provide
information

* about the cohort member(s) in both sweeps (either as Main or
Partner respondent).

SAVE OUTFILE =
'mcs_working_folder/mcs5_mcs6_parent_cm_interview_by_PNUM.sav'
.
```

* Example code L

```
* --------------------.

* Create a composite variable per child in parent_cm dataset.

* ----------------------------------------------------.

* this syntax creates a mean of Main and Partner responses on
ASLU variable .

* You can use other functions instead of MEAN, like SD, MIN,
MAX, etc.

GET FILE = 'mcs6_folder/mcs6_parent_cm_interview.sav'.

AGGREGATE

  outfile=*

  overwrite=yes
```

```
  mode=addvariables

  /break= MCSID FCNUM00

  /composite_ASLU = MEAN(FPASLU00)

  /groupsize = N.

VARIABLE LABELS composite_ASLU 'Mean score of ASLU assessment
by Main/Partner respondent(s) of the cohort member'.

VARIABLE LABELS groupsize 'Number of respondents (Main only,
Partner only, or Main&Partner) providing information in ASLU'.

* Let's take a look at the result .

TEMPORARY.

SELECT IF ((MCSID EQ 'M10002P') OR (MCSID EQ 'M10611J') OR
(MCSID EQ 'M10106W') OR (MCSID EQ 'M10063C')).

SUMMARIZE

/TABLES = MCSID

FPNUM00

FELIG00

FRESP00

FCNUM00

FPASLU00

composite_ASLU

groupsize

/FORMAT = VALIDLIST NOCASENUM TOTAL
```

```
/TITLE = 'Overview of specific cases on composite score for
each Cohort Member (groupsize = number of parents ELIG
providing information)'

/MISSING = VARIABLE

/CELLS = COUNT.
```

* Example code M

```
* --------------------.


* Merging datasets of different structures to the
mcs_longitudinal_family_file .

* ------------------------------------------------------------
-----------------------------------------------.

* we do not need to sort files as they are all sorted by
MCSID,

* however, if you have worked on the file before make sure you

* sort it before this step.

* merge with a _parent_ level dataset .

GET FILE = "mcs6_folder/mcs6_parent_derived.sav".

MATCH FILES /FILE = *

/IN source_parent

/TABLE = "mcs6_folder/mcs_longitudinal_family_file.sav"

/IN = source_longitudinal_file
```

```
/BY MCSID .

EXECUTE.

CROSSTABS source_parent by source_longitudinal_file.


* merge with a _cm_ level dataset .

GET FILE = "mcs6_folder/mcs6_cm_derived.sav".

MATCH FILES /FILE = *

/IN source_cm

/TABLE = "mcs6_folder/mcs_longitudinal_family_file.sav"

/IN = source_longitudinal_file

/BY MCSID .

EXECUTE.

CROSSTABS source_cm by source_longitudinal_file.


* merge with a _family_ level dataset .

GET FILE = "mcs6_folder/mcs6_family_derived.sav".

MATCH FILES /FILE = *

/IN source_familyDV

/TABLE = "mcs6_folder/mcs_longitudinal_family_file.sav"

/IN = source_longitudinal_file

/BY MCSID .

EXECUTE.
```

```
CROSSTABS source_familyDV by source_longitudinal_file.


* merge with a _parent_cm_ level dataset .

GET FILE = "mcs6_folder/mcs6_parent_cm_interview.sav".

MATCH FILES /FILE = *

/IN source_parent_cm

/TABLE = "mcs6_folder/mcs_longitudinal_family_file.sav"

/IN = source_longitudinal_file

/BY MCSID .

EXECUTE.

CROSSTABS source_parent_cm by source_longitudinal_file.
```

* Example code N

```
* --------------------.

* Example I - merging two 1-level datasets that have different
identifiers.

* Please run example code H to get the _parent_ level dataset
below.

GET FILE =
"mcs_working_folder/mcs6_parent_plus_proxy_interview.sav".

SORT CASES BY MCSID (A).
```

```
* create a composite score of the general health of the Main &
Partner.

AGGREGATE

  outfile=*

  overwrite=yes

  mode=addvariables

  /break= MCSID

  /composite_HEALTH = MEAN(FPGENA00)

  /groupsize = N.

VARIABLE LABELS composite_HEALTH 'Mean self-assessed health of
Main/Partner respondent(s) in each family'.

VARIABLE LABELS groupsize 'Number of respondents (Main only,
Partner only, or Main&Partner) with information on GENA'.

* Let's take a look at the result .

TEMPORARY.

SELECT IF ((MCSID EQ 'M10002P') OR (MCSID EQ 'M10611J') OR
(MCSID EQ 'M10451L') OR (MCSID EQ 'M10106W') OR (MCSID EQ
'M10063C')).

SUMMARIZE

/TABLES = MCSID

FELIG00

FRESP00

FPGENA00
```

```
composite_HEALTH

groupsize

/FORMAT = VALIDLIST NOCASENUM TOTAL

/TITLE = 'Overview of specific cases on composite score for
each family (mean Health)'

/MISSING = VARIABLE

/CELLS = COUNT.

* at this stage we can just keep the one row for each family,
however, by

* restructuring in wide format (one row per family) we keep
all the information

* that we may need for crosschecking.


* restructure the datasets into wide format (one row per
family).

FREQUENCIES FELIG00 .

* We create a Main / Partner only ELIG.

IF (FELIG00 EQ 1) ELIG = 1.

IF (FELIG00 EQ 2 OR FELIG00 EQ 3) ELIG = 2.

EXECUTE.

VARIABLE LABELS ELIG 'Eligibility Partner collapsed'.

VALUE LABELS ELIG 1 'Main' 2 'Partner / Proxy'.

MISSING VALUES ALL ().
```

```
CROSSTABS FELIG00 BY ELIG .

* resctructure using ELIG.

CASESTOVARS

/ID = MCSID

/INDEX = ELIG.

* Let's look at the new dataset.

TEMPORARY.

SELECT IF ((MCSID EQ 'M10002P') OR (MCSID EQ 'M10611J') OR
(MCSID EQ 'M10451L') OR (MCSID EQ 'M10106W') OR (MCSID EQ
'M10063C')).

LIST.

* the variables containing the information of the Main have
been suffixed with 1 and of the Partner with 2.

SORT CASES BY MCSID (A).

SAVE OUTFILE =
"mcs_working_folder/mcs6_parent_plus_proxy_interview_wide.sav"
.


* merge _cm_ level with the wide restructured parent dataset
(one row per family).

GET FILE = "mcs6_folder/mcs6_cm_interview.sav".

MATCH FILES /FILE = *

/IN source_cm
```

```
/TABLE =
"mcs_working_folder/mcs6_parent_plus_proxy_interview_wide.sav"

/IN = source_parent_wide

/BY MCSID .

EXECUTE.

CROSSTABS source_cm by source_parent_wide.

* for some families there is no parent interview but the CM
has participated.


* comparison between parents' general health and CM's.

CROSSTABS composite_HEALTH by FCCGHE00 .
```

* Example code O

```
* --------------------.

* Example II - merging a 1-level dataset (_cm_) with a 2-level
dataset (_parent_cm_) resulting into a 1-level structure
(_cm_).

* Please check example code L to get the _parent_cm_ level
dataset, it is similar.

GET FILE = 'mcs6_folder/mcs6_parent_cm_interview.sav'.

* we generate a composite measure of ASLU like in the example
code L.

AGGREGATE
```

```
  outfile=*

  overwrite=yes

  mode=addvariables

  /break= MCSID FCNUM00

  /composite_ASLU = MEAN(FPASLU00)

  /groupsize = N.

VARIABLE LABELS composite_ASLU 'Mean score of ASLU assessment
by Main/Partner respondent(s) of the cohort member'.

VARIABLE LABELS groupsize 'Number of respondents (Main only,
Partner only, or Main&Partner) providing information in ASLU'.

* Let's take a look at the result .

TEMPORARY.

SELECT IF ((MCSID EQ 'M10002P') OR (MCSID EQ 'M10611J') OR
(MCSID EQ 'M10106W') OR (MCSID EQ 'M10063C')).

SUMMARIZE

/TABLES = MCSID

FPNUM00

FELIG00

FRESP00

FCNUM00

FPASLU00

composite_ASLU

groupsize
```

```
/FORMAT = VALIDLIST NOCASENUM TOTAL

/TITLE = 'Overview of specific cases on composite score for
each Cohort Member (groupsize = number of parents ELIG
providing information)'

/MISSING = VARIABLE

/CELLS = COUNT.


SORT CASES BY MCSID (A).

SAVE OUTFILE =
'mcs_working_folder/mcs6_parent_cm_interview_reduced.sav'

/KEEP

MCSID

FPNUM00

FELIG00

FRESP00

FCNUM00

FCCREL00

FPASLU00

composite_ASLU

groupsize.


* we turn the _parent_cm_ dataset into wide format (one row
per CM).
```

```
GET FILE =
'mcs_working_folder/mcs6_parent_cm_interview_reduced.sav'.

FREQUENCIES FELIG00 .

* We create a Main / Partner only ELIG.

IF (FELIG00 EQ 1) ELIG = 1.

IF (FELIG00 EQ 2 OR FELIG00 EQ 3) ELIG = 2.

EXECUTE.

VARIABLE LABELS ELIG 'Eligibility Partner collapsed'.

VALUE LABELS ELIG 1 'Main' 2 'Partner / Proxy'.

MISSING VALUES ALL ().

CROSSTABS FELIG00  BY ELIG .

* resctructure using ELIG.

CASESTOVARS

/ID = MCSID FCNUM00

/INDEX = ELIG.

* Let's look at the new dataset.

TEMPORARY.

SELECT IF ((MCSID EQ 'M10002P') OR (MCSID EQ 'M10611J') OR
(MCSID EQ 'M10451L') OR (MCSID EQ 'M10106W') OR (MCSID EQ
'M10063C')).

LIST.

* the variables containing the information of the Main have
been suffixed with 1 and of the Partner with 2.
```

```
* there is one row per child, so the dataset is on the _cm_
level.

SORT CASES BY MCSID (A).

SAVE OUTFILE =
"mcs_working_folder/mcs6_parent_cm_interview_reduced_wide_on_c
m_level.sav".


* merge _cm_interview level with the wide restructured parent
dataset (one row per family).

GET FILE = "mcs6_folder/mcs6_cm_interview.sav".

MATCH FILES /FILE = *

/IN source_cm

/TABLE =
"mcs_working_folder/mcs6_parent_cm_interview_reduced_wide_on_c
m_level.sav"

/IN = source_parent_cm_wide

/BY MCSID FCNUM00 .

EXECUTE.

CROSSTABS source_cm by source_parent_cm_wide.

* for some families there is no parent interview but the CM
has participated.


* comparison between parents' perception of CM's likelihood to
go to the University and CM's perception on the same topic.

CROSSTABS FCSTYU00 by composite_ASLU .
```

* Example code P

```
* --------------------.

* Example III - merging a 2-level dataset (_parent_cm_) with a
1-level dataset (_cm_) resulting into a 2-level dataset
(_cm_).

GET FILE = 'mcs6_folder/mcs6_parent_cm_interview.sav'.

MATCH FILES /FILE = *

/IN source_cm

/TABLE = "mcs6_folder/mcs6_cm_interview.sav"

/IN = source_parent_cm

/BY MCSID FCNUM00 .

EXECUTE.

CROSSTABS source_cm by source_parent_cm.


CROSSTABS FCTRST0A by FPSCHC00.
```

STATA syntax

* Setting up folders in STATA

```
global mcs5_folder "\user_folder_path\mcs5_"
global mcs6_folder "\user_folder_path\mcs6_"
global mcs_working_folder
"\user_folder_path\mcs_working_folder"
```

* Example code A

```
* --------------------.
* Overview of the hhgrid .
* -------------------------.
use "$mcs6_folder/mcs6_hhgrid.dta", clear
* Overview of PRES / CREL / multiple Cohort Members per
family.
gen example_families = 1 if MCSID == "M10002P" | MCSID ==
"M10063C" | MCSID == "M10611J" ///
| MCSID == "M10106W" | MCSID == "M10063C"
tab example_families

ds MCSID FPNUM00 FELIG00 FRESP00 FCNUM00 FHCREL00 FHPRES00
return list
local varlist = r(varlist)
list `varlist' if example_families == 1, divider sep(4)
```

* Example code B

```
* --------------------.
* Concatenating MCSID & person identifier to get a unique
person identifier .
* ------------------------------------------------------------
-------------------------------------------.
use "$mcs6_folder/mcs6_hhgrid.dta", clear

* a Person ID for each adult in the household (excluding
cohort members) .
gen PnumID = MCSID + string(FPNUM00) if (FPNUM00 > 0)

* a Person ID for each individual of the household (Cohort
Member or other person) .
gen CMrow = "_C1" if FCNUM00 == 1
```

```
replace CMrow = "_C2" if FCNUM00 == 2
replace CMrow = "_C3" if FCNUM00 == 3
gen PID = MCSID + "_" + string(FPNUM00) if (FPNUM00 > 0)
replace PID = MCSID + CMrow if (FCNUM00 > 0)
```

* Example code C

```
* --------------------.
* Overview of the _family_derived .
use "$mcs6_folder/mcs6_family_derived.dta", clear
tab FDNOCM00
tab FDRSPO00

* Overview of the _cm_derived .
use "$mcs6_folder/mcs6_cm_derived.dta", clear
tab FCNUM00
```

* Example code D

```
* --------------------.
* Overview of _parent_derived .
* ---------------------------------------.
use "$mcs6_folder/mcs6_parent_derived.dta", clear
gen example_families = 1 if MCSID == "M10002P" | MCSID ==
"M10041W" | MCSID == "M23136V" | MCSID == "M10106W" | MCSID ==
"M10063C".
tab example_families

ds MCSID FPNUM00 FELIG00 FRESP00
return list
local varlist = r(varlist)
list `varlist' if example_families == 1, divider sep(4)
```

* Example code E

```
* --------------------.
* merge _parent_ structure datasets from different sweeps.
* ------------------------------------------------------------
----------------------.

*  ~ ~ ~ ~ ~ ~ merge the two datasets on MCSID & ELIG  ~ ~ ~ ~
~ ~ .
* prepare the mcs6_parent dataset.
use "$mcs6_folder/mcs6_parent_interview.dta", clear
tab FPNUM00
tab FELIG00
gen ELIG = FELIG00
```

```
save
"$mcs_working_folder/mcs6_parent_interview_cross_sweep_merging
.dta", replace

* prepare the mcs5_parent_dataset.
use "$mcs5_folder/mcs5_parent_interview.dta", clear
tab EPNUM00
tab EELIG00
gen ELIG = EELIG00
save
"$mcs_working_folder/mcs5_parent_interview_cross_sweep_merging
.dta", replace

* merge .
use
"$mcs_working_folder/mcs6_parent_interview_cross_sweep_merging
.dta", clear
merge 1:1 MCSID ELIG using
"$mcs_working_folder/mcs5_parent_interview_cross_sweep_merging
.dta"
* Outcome perusal: Main and Partner respondents (ELIG) in both
sweeps.
tab ELIG
tab FELIG00 EELIG00
* Let us see how many of the merged Main & Partner respondents
(ELIG) have
* the same PNUM, therefore they are they same person .
* Values in the diagonal of the crosstabulation are
respondents who are the
* same in both sweeps.
tab FPNUM0 EPNUM00 if (ELIG == 1)
gen same_respondent = 1 if (FPNUM00 == EPNUM00)
replace same_respondent = 0 if (same_respondent != 1)
label variable same_respondent "Is the Main/Partner respondent
the same btw S5 & S6?"
label define same_respondent 1 "Same" 0 "Different or missing
data"
tab same_respondent
tab ELIG same_respondent
save
"$mcs_working_folder/mcs5_mcs6_parent_interview_by_ELIG.dta",
replace


*  ~ ~ ~ ~ ~ ~ merge the two datasets on MCSID & PNUM ~ ~ ~ ~
~ ~ .
* prepare the mcs6_parent_ dataset.
use "$mcs6_folder/mcs6_parent_interview.dta", clear
gen PNUM = FPNUM00
```

```
save
"$mcs_working_folder/mcs6_parent_interview_cross_sweep_merging
.dta", replace
* prepare the mcs5_parent_dataset.
use "$mcs5_folder/mcs5_parent_interview.dta", clear
gen PNUM = EPNUM00
save
"$mcs_working_folder/mcs5_parent_interview_cross_sweep_merging
.dta", replace

* merge.
use
"$mcs_working_folder/mcs6_parent_interview_cross_sweep_merging
.dta", clear
merge m:1 MCSID PNUM using
"$mcs_working_folder/mcs5_parent_interview_cross_sweep_merging
.dta"
* Outcome perusal: 17.211 parents/carers (PNUM) are in both
sweeps.
* Let us see how many of the merged parents/carers respondents
(PNUM) have
* the same role in the interview (ELIG: Main or Partner) .
gen same_eligibility = 1 if (FELIG00 == EELIG00)
replace same_eligibility = 0 if (same_eligibility != 1)
label variable same_eligibility "Is the role at the interview
(ELIG) the same btw S5 & S6?"
label define same_eligibility 1 "Same" 0 "Different or missing
data"
tab same_eligibility
tab PNUM same_eligibility
save
"$mcs_working_folder/mcs5_mcs6_parent_interview_by_PNUM.dta",
replace
```

* Example code F

```
* --------------------.
* Create a composite variable per family in the _parent_
structure file .
* ------------------------------------------------------------
-------------------------------------.
* this syntax creates a mean of Main and Partner responses on
GENA variable .
* You can use other functions instead of mean, like
use "$mcs6_folder/mcs6_parent_interview.dta", clear
egen composite_GENA = mean(FPGENA00), by (MCSID)
egen groupsize = count(FPGENA00), by (MCSID)
label variable composite_GENA "Mean score of GENA of
Main/Partner respondent(s) per family"
```

```
label variable groupsize "Number of respondents (Main only,
Partner only, or Main&Partner) providing information in GENA"
* Let's take a look at the result .
gen example_families = 1 if MCSID == "M10002P" | MCSID ==
"M10611J" | MCSID == "M10106W" | MCSID == "M10063C".
tab example_families
* run this code with/without nolabel to hide/see the value
labels
ds MCSID FPNUM00 FELIG00 FRESP00 FPGENA00 composite_GENA
groupsize
return list
local varlist = r(varlist)
list `varlist' if example_families == 1, divider sep(4)
nolabel

* this syntax selects the higher NVQ of Main and Partner
respondents.
* You can use other functions instead of MEAN, like sd, min,
max etc.
use "$mcs6_folder/mcs6_parent_derived.dta", clear
tab FDNVQ00, nolabel
gen NVQ = FDNVQ00
replace NVQ = . if (NVQ < 1)
replace NVQ = . if (NVQ > 5)
egen composite_NVQ = max(NVQ), by (MCSID)
egen groupsize = count(FPNUM00), by (MCSID)
label variable composite_NVQ "Highest NVQ of Main/Partner
respondent(s) in each family"
label variable groupsize "Number of respondents (Main only,
Partner only, or Main&Partner) with information on NVQ"
* Let's take a look at the result .
gen example_families = 1 if MCSID == "M10002P" | MCSID ==
"M10611J" | MCSID == "M10451L" | MCSID == "M10106W" | MCSID ==
"M10063C".
tab example_families
* run this code with/without nolabel to hide/see the value
labels
ds MCSID FPNUM00 FELIG00 FRESP00 FDNVQ00 NVQ composite_NVQ
groupsize
return list
local varlist = r(varlist)
list `varlist' if example_families == 1, divider sep(4)
nolabel
```

* Example code G

```
* --------------------.
* Overview of _proxy_partner_interview .
* ----------------------------------------------------.
```

```
use "$mcs6_folder/mcs6_proxy_partner_interview.dta", clear
tab FXCREL00
tab FXCREL00 FXPSEX00
tab FXPXRE00 FXPXIN00
```

* Example code H

```
* --------------------.
* Combining proxy_partner_interview with parent_interview .
* ------------------------------------------------------------
----------------------.
* we keep only rows where the Main agreed to provide
information about the non-available Partner.
keep if FXPXIN00 == 1
tab FXPXGE00
* we rename the variable to the variable name that is used in
the parent_interview dataset.
rename FXPXGE00 FPGENA00
keep MCSID FELIG00 FRESP00 FXCREL00 FPGENA00
save
"$mcs_working_folder/mcs6_proxy_partner_interview_for_connecti
on_to_parent.dta", replace

* open the parent_interview dataset and keep only the
variables needed .
use "$mcs6_folder/mcs6_parent_interview.dta", clear
tab FPGENA00
keep MCSID FELIG00 FRESP00 FPCREL00 FPGENA00
save
"$mcs_working_folder/mcs6_parent_interview_for_connection_to_p
roxy.dta", replace

* open the reduced parent_interview dataset.
use
"$mcs_working_folder/mcs6_parent_interview_for_connection_to_p
roxy.dta", clear
* we add cases/rows to the dataset.
append using
"$mcs_working_folder\mcs6_proxy_partner_interview_for_connecti
on_to_parent.dta"
tab FPGENA00
tab FPGENA00 FELIG00
save
"$mcs_working_folder/mcs6_parent_plus_proxy_interview.dta",
replace
```

* Example code I

```
* --------------------.
```

```
* merge _cm_ structure datasets from different sweeps.
* ------------------------------------------------------------
---------------.
* prepare the mcs6_cm dataset.
use "$mcs6_folder/mcs6_cm_interview.dta", clear
tab FCNUM00
gen CNUM = FCNUM00
save
"$mcs_working_folder/mcs6_cm_interview_cross_sweep_merging.dta
", replace

* prepare the mcs5_cm_dataset.
use "$mcs5_folder/mcs5_cm_interview.dta", clear
tab ECNUM00
gen CNUM = ECNUM00
save
"$mcs_working_folder/mcs5_cm_interview_cross_sweep_merging.dta
", replace

* merge the two datasets on MCSID & CNUM .
use
"$mcs_working_folder/mcs6_cm_interview_cross_sweep_merging.dta
", clear
merge 1:1 MCSID CNUM using
"$mcs_working_folder/mcs5_cm_interview_cross_sweep_merging.dta
"
* Outcome perusal: cohort members in both datasets.
tab CNUM
tab FCNUM00 ECNUM00
save "$mcs_working_folder/mcs5_mcs6_cm_interview.dta", replace
```

* Example code J

```
* --------------------.
* Overview of _parent_cm_interview .
* ------------------------------------------------.
* Parent's interview about the CM(s) of the household.
use "$mcs6_folder/mcs6_parent_cm_interview.dta", clear
tab FCNUM00
*
gen example_families = 1 if MCSID == "M10002P" | MCSID ==
"M10611J" | MCSID == "M10106W" | MCSID == "M10063C"
tab example_families
* run this code with/without nolabel to hide/see the value
labels
ds MCSID FPNUM00 FELIG00 FCNUM00 FCCSEX00 FCCAGE00 FPSDPF00
FPASLU00
return list
local varlist = r(varlist)
```

```
list `varlist' if example_families == 1, divider sep(4)
nolabel
```

* Example code K

```
* --------------------.
* Merge _parent_cm_ level datasets between sweeps.
* ------------------------------------------------------------
--------------.
* Create row identifier to connect for _parent_cm_ datasets.
* ROWid = (ELIG or PNUM) + (CNUM Child 1/2/3) .

* ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~  ELIG + CNUM = ROWid ~ ~ ~ ~ ~ ~ ~ ~
~ ~ ~ ~ .
* prepare mcs6_parent_cm_interview .
use "$mcs6_folder/mcs6_parent_cm_interview.dta", clear
gen ELIGrow = "M" if (FELIG00 == 1)
replace ELIGrow = "P" if (FELIG00 == 2)
gen CNUM = FCNUM00
tab CNUM ELIGrow
gen ROWid = ELIGrow + string(CNUM)

* Check that the ROWid matches the crosstabulation of ELIG &
CNUM .
tab ROWid
save
"$mcs_working_folder/mcs6_parent_cm_interview_cross_sweep_merg
ing.dta", replace

* prepare mcs5_parent_cm_interview .
use "$mcs5_folder/mcs5_parent_cm_interview.dta", clear
gen ELIGrow = "M" if (EELIG00 == 1)
replace ELIGrow = "P" if (EELIG00 == 2)
gen CNUM = ECNUM00
tab CNUM ELIGrow
gen ROWid = ELIGrow + string(CNUM)
* Check that the ROWid matches the crosstabulation of ELIG &
CNUM .
tab ROWid
save
"$mcs_working_folder/mcs5_parent_cm_interview_cross_sweep_merg
ing.dta", replace
* Merge .
use
"$mcs_working_folder/mcs6_parent_cm_interview_cross_sweep_merg
ing.dta"
merge 1:1 MCSID ROWid using
"$mcs_working_folder/mcs5_parent_cm_interview_cross_sweep_merg
ing.dta"
```

```
* Outcome perusal: parents (Main/Partner providing information
about
* each of the cohort members) are in both sweeps.
save
"$mcs_working_folder/mcs5_mcs6_parent_cm_interview_by_ELIG.dta
", replace


* ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~  PNUM + CNUM = ROWid ~ ~ ~ ~ ~ ~ ~ ~ ~
~ ~ ~ ~ .
* prepare mcs6_parent_cm_interview .
use "$mcs6_folder/mcs6_parent_cm_interview.dta", clear
gen CMrow = "_C1" if (FCNUM00 == 1)
replace CMrow = "_C2" if (FCNUM00 == 2)
replace CMrow = "_C3" if (FCNUM00 == 3)
gen Prow = "P" + string(FPNUM00)
gen ROWid = Prow + CMrow
* Let's check that the totals match.
tab FPNUM00 FCNUM00
tab ROWid
save
"$mcs_working_folder/mcs6_parent_cm_interview_cross_sweep_merg
ing.dta", replace

* prepare mcs5_parent_cm_interview .
use "$mcs5_folder/mcs5_parent_cm_interview.dta", clear
gen CMrow = "_C1" if (ECNUM00 == 1)
replace CMrow = "_C2" if (ECNUM00 == 2)
replace CMrow = "_C3" if (ECNUM00 == 3)
gen Prow = "P" + string(EPNUM00)
gen ROWid = Prow + CMrow
* Let's check that the totals match.
tab EPNUM00 ECNUM00
tab ROWid
save
"$mcs_working_folder/mcs5_parent_cm_interview_cross_sweep_merg
ing.dta", replace

* Merge .
use
"$mcs_working_folder/mcs6_parent_cm_interview_cross_sweep_merg
ing.dta", clear
merge 1:1 MCSID ROWid using
"$mcs_working_folder/mcs5_parent_cm_interview_cross_sweep_merg
ing.dta"
* Outcome perusal: Individuals (PNUM) provide information
* about the cohort member(s) in both sweeps (either as Main or
Partner respondent).
```

```
save
"$mcs_working_folder/mcs5_mcs6_parent_cm_interview_by_PNUM.dta
", replace
```

* Example code L

```
* --------------------.
* Create a composite variable per child in parent_cm dataset.
* ---------------------------------------------------.
* this syntax creates a mean of Main and Partner responses on
ASLU variable .
* You can use other functions instead of MEAN, like sd, min,
max etc.
use "$mcs6_folder/mcs6_parent_cm_interview.dta", clear
tab FPASLU00
gen ASLU = FPASLU00
replace ASLU = . if (FPASLU00 <1)
egen composite_ASLU = mean(ASLU), by (MCSID FCNUM00)
egen groupsize = count(FPNUM00), by (MCSID FCNUM00)
label variable composite_ASLU "Mean score of ASLU assessment
by Main/Partner respondent(s) of the cohort member"
label variable groupsize "Number of respondents (Main only,
Partner only, or Main&Partner) providing information in ASLU"
* Let's take a look at the result .
gen example_families = 1 if MCSID == "M10002P" | MCSID ==
"M10611J" | MCSID == "M10106W" | MCSID == "M10063C".
tab example_families
* Overview of specific cases on composite score for each
Cohort Member (groupsize = number of parents ELIG providing
information)
* run this code with/without nolabel to hide/see the value
labels
ds MCSID FPNUM00 FELIG00 FCNUM00 FPASLU00 ASLU composite_ASLU
groupsize
return list
local varlist = r(varlist)
list `varlist' if example_families == 1, divider sep(4)
nolabel
```

* Example code M

```
* --------------------.

* Merging datasets of different structures to the
mcs_longitudinal_family_file .
* ---------------------------------------------------------
-----------------------------------------------.
```

```
* we do not need to sort files as they are all sorted by
MCSID,
* however, if you have worked on the file before make sure you
* sort it before this step.
* merge with a _parent_ level dataset .
use "$mcs6_folder/mcs6_parent_derived.dta", clear
merge m:1 MCSID using
"$mcs6_folder/mcs_longitudinal_family_file.dta"

* merge with a _cm_ level dataset .
use "$mcs6_folder/mcs6_cm_derived.dta", clear
merge m:1 MCSID using
"$mcs6_folder/mcs_longitudinal_family_file.dta"

* merge with a _family_ level dataset .
use "$mcs6_folder/mcs6_family_derived.dta", clear
merge m:1 MCSID using
"$mcs6_folder/mcs_longitudinal_family_file.dta"

* merge with a _parent_cm_ level dataset .
use "$mcs6_folder/mcs6_parent_cm_interview.dta", clear
merge m:1 MCSID using
"$mcs6_folder/mcs_longitudinal_family_file.dta"
```

* Example code N

```
* --------------------.
* Example I - merging two 1-level datasets that have different
identifiers.
* Please run example code H to get the _parent_ level dataset
below.
use
"$mcs_working_folder/mcs6_parent_plus_proxy_interview.dta",
clear

* restructure the datasets into wide format (one row per
family).
tab FELIG00
* We create a Main / Partner only ELIG.
gen ELIG_suffix = "_M_" if (FELIG00 == 1) // Main
replace ELIG_suffix = "_P_" if (FELIG00 == 2) //(Proxy)
Partner
replace ELIG_suffix = "_P_" if (FELIG00 == 3) //(Proxy)
Partner
* resctructure using ELIG.
reshape wide FELIG00 FRESP00 FPCREL00 FPGENA00 FXCREL00 ,
i(MCSID) j(ELIG_suffix) string
* the variables containing the information of the Main have
been suffixed with M and of the (Proxy) Partner with P.
```

```
save
"$mcs_working_folder/mcs6_parent_plus_proxy_interview_wide.dta
", replace

* merge _cm_ level with the wide restructured parent dataset
(one row per family).
use "$mcs6_folder/mcs6_cm_interview.dta", clear
merge m:1 MCSID using
"$mcs_working_folder/mcs6_parent_plus_proxy_interview_wide.dta
"
* for some families there is no parent interview but the CM
has participated.

* create a composite score of the general health of the Main &
Partner.
gen GENA_M = FPGENA00_M_
replace GENA_M = . if (GENA_M < 0)
gen GENA_P = FPGENA00_P_
replace GENA_P = . if (GENA_P < 0)
gen composite_GENA = (GENA_M + GENA_P)/2
* Let's take a look at the result .
gen example_families = 1 if MCSID == "M10002P" | MCSID ==
"M10611J" | MCSID == "M10106W" | MCSID == "M10063C".
tab example_families
* Overview of specific cases on composite score for each
family (mean Health)
* run this code with/without nolabel to hide/see the value
labels
ds MCSID GENA_M GENA_P composite_GENA
return list
local varlist = r(varlist)
list `varlist' if example_families == 1, divider sep(4)
nolabel
gen CGHE = FCCGHE00
replace CGHE = . if (CGHE < 0)
* comparison between parents' general health and CM's.
tabulate composite_GENA CGHE
```

* Example code O

```
* --------------------.
* Example II - merging a 1-level dataset (_cm_) with a 2-level
dataset (_parent_cm_) resulting into a 1-level structure
(_cm_).
* Please check example code L to get the _parent_cm_ level
dataset, it is similar.
use "$mcs6_folder/mcs6_parent_cm_interview.dta", clear
keep MCSID FPNUM00 FELIG00 FRESP00 FCNUM00 FCCREL00 FPASLU00
```

```
save
"$mcs_working_folder/mcs6_parent_cm_interview_reduced.dta",
replace

* we turn the _parent_cm_ dataset into wide format (one row
per CM).
use
"$mcs_working_folder/mcs6_parent_cm_interview_reduced.dta",
clear
* restructure the datasets into wide format (one row per
family).
tab FELIG00
* We create a Main / Partner only ELIG.
gen ELIG_suffix = "_M_" if (FELIG00 == 1) // Main
replace ELIG_suffix = "_P_" if (FELIG00 == 2) //(Proxy)
Partner
replace ELIG_suffix = "_P_" if (FELIG00 == 3) //(Proxy)
Partner
* resctructure using ELIG.
reshape wide FPNUM00 FELIG00 FRESP00 FCCREL00 FPASLU00 ,
i(MCSID FCNUM00) j(ELIG_suffix) string
gen ASLU_M = FPASLU00_M_
replace ASLU_M = . if (ASLU_M < 0)
gen ASLU_P = FPASLU00_P_
replace ASLU_P = . if (ASLU_P < 0)
gen composite_ASLU = (ASLU_M + ASLU_P)/ 2
* impute the values of the one parent or the other if one is
missing and composite
* score has not been calculated - please adjust the
calculation of the composite score
* to the needs of your project
replace composite_ASLU = ASLU_M if missing(ASLU_P)
replace composite_ASLU = ASLU_P if missing(ASLU_M)
tab composite_ASLU

* Let's take a look at the result .
gen example_families = 1 if MCSID == "M10002P" | MCSID ==
"M10611J" | MCSID == "M10106W" | MCSID == "M10063C".
tab example_families
* Mean score of ASLU assessment by Main/Partner respondent(s)
of the cohort member
* run this code with/without nolabel to hide/see the value
labels
ds MCSID FPNUM00_M_ FPNUM00_P_ FCNUM00 FPASLU00_M_ FPASLU00_P_
composite_ASLU
return list
local varlist = r(varlist)
list `varlist' if example_families == 1, divider sep(4)
nolabel
```

```
save
"$mcs_working_folder/mcs6_parent_cm_interview_reduced_wide_on_
cm_level.dta", replace

* merge _cm_interview level with the wide restructured parent
dataset (one row per family).
use "$mcs6_folder/mcs6_cm_interview.dta", clear
merge 1:1 MCSID FCNUM00 using
"$mcs_working_folder/mcs6_parent_cm_interview_reduced_wide_on_
cm_level.dta"
* for some families there is no parent interview but the CM
has participated.

* comparison between parents' perception of CM's likelihood to
go to the University and CM's perception on the same topic.
tab FCSTYU00 composite_ASLU
twoway lfitci composite_ASLU FCSTYU00  || scatter
composite_ASLU FCSTYU00
```

* Example code P

```
* --------------------.
* Example III - merging a 2-level dataset (_parent_cm_) with a
1-level dataset (_cm_) resulting into a 2-level dataset
(_cm_).
use "$mcs6_folder/mcs6_parent_cm_interview.dta", clear
merge m:1 MCSID FCNUM00 using
"$mcs6_folder/mcs6_cm_interview.dta"
tab FCTRST0A FPSCHC00

use "$mcs6_folder/mcs6_cm_interview.dta", clear
merge 1:m MCSID FCNUM00 using
"$mcs6_folder/mcs6_parent_cm_interview.dta"
tab FCTRST0A FPSCHC00
```