SHARE
Survey of Health, Ageing and Retirement in Europe
50+ in Europe

Generating *easy*SHARE
Guidelines, Structure, Content and Programming

Stefan Gruber, Christian Hunkler and Stephanie Stuck

Working Paper Series 17-2014

European Research Area
SIXTH FRAMEWORK PROGRAMME
SEVENTH FRAMEWORK PROGRAMME
Federal Ministry of Education and Research

Generating *easy*SHARE

Guidelines, Structure, Content and Programming

Stefan Gruber[a], Christian Hunkler[a]
& Stephanie Stuck[a]

February 5, 2014

Abstract

This paper provides an overview of the construction, structure and content of *easy*SHARE – a simplified dataset that is based on the scientific release of the Survey of Health, Ageing and Retirement in Europe (SHARE). *easy*SHARE contains a subset of ready-to-analyse variables and indices. It is designed for student training on country-comparative and longitudinal analyses as well as for introducing SHARE to researchers who have little experience in quantitative analyses of complex survey data. This paper puts special emphasis on the extraction and generation of the dataset using the statistical software package Stata. We also provide an example of how to extract more variables from the scientific release of SHARE to include them in the *easy*SHARE dataset.

Acknowledgement

We are grateful to Agnes Orban who was part of the *easy*SHARE team until end of 2011. We also thank Axel Börsch-Supan and Martina Brandt for their efforts to secure funding for *easy*SHARE. Valuable contributions have been made by Radim Bohacek, Dimitris Christelis, Danilo Cavapozzi, Christian Deindl, Giuseppe De Luca, Thorsten Kneip, George Papadoudis, Giacomo Pasini, Mauricio Avendano Pabon, Morten Schuth and Morten Wahrendorf.

[a] Munich Center for the Economics of Aging (MEA), Max Planck Institute for Social Law and Social Policy

1. Introduction

The Survey of Health, Ageing and Retirement in Europe (SHARE[1]) is a rich database that provides information on health, socio-economic status and social and family networks of more than 85,000 individuals aged 50 and over (Börsch-Supan et al. 2013a). Up to the present, SHARE has released three regular panel waves (2004, 2006, 2010) and one wave on retrospective life histories (2008, SHARELIFE). For more details on SHARELIFE see Börsch-Supan et al. (2013b) and Schröder (2011). Further information on the methodology and content of the regular panel waves is provided by Börsch-Supan and Jürges (2005), Börsch-Supan et al. (2005, 2008, 2011 and 2013c) and Malter and Börsch-Supan (2013).

Due to its richness SHARE is also complex. The multitude of countries participating in SHARE with their institutional variety and different languages, combined with a large interdisciplinary set of variables stored in more than 100 single data modules, and the differentiation of individual-, couple- and household-level data, make SHARE a complicated database. In contrast the *easy*SHARE file is a single simplified dataset for student training and for researchers who are less experienced in the quantitative analysis of complex panel data.

*easy*SHARE includes the same number of observations as the main release of SHARE but is restricted to a subset of variables. It contains the regular panel waves of SHARE (wave 1, wave 2 and wave 4) and some information collected in SHARELIFE (wave 3). This allows storing *easy*SHARE as one single data file without the need for complex merging of waves and modules. The file includes selected information from the SHARE CAPI (Computer Assisted Personal Interview) modules and in some instances we use measures collected in the the drop-off questionnaires. We also incorporate variables from the "generated variables datasets" and create additional "easy to use"-variables. Additionally, we put special emphasis on comparability with the US Health and Retirement Study (HRS).

To facilitate analyzing the data we transfer the information that was collected on couple- and household-level to the individual level. Several generated variables, such as indices and recoded health, demographic, social and economic measures allow analyses without the need for extensive data preparation. Furthermore, we add a series of additional missing value codes which give detailed information on why exactly information is missing.

This paper gives an overview of the *easy*SHARE dataset. First we summarize the guidelines that *easy*SHARE is based on, followed by a description of the data structure. The next section depicts the subset of variables that the dataset contains. Section 5 explains the general

[1] This paper uses data from SHARE wave 4 release 1.1.1, as of March 28th 2013 and SHARE wave 1 and 2 release 2.6.0, as of November 29th 2013 and SHARELIFE release 1, as of November 24th 2010. The SHARE data collection has been primarily funded by the European Commission through the 5th Framework Programme (project QLK6-CT-2001-00360 in the thematic programme Quality of Life), through the 6th Framework Programme (projects SHARE-I3, RII-CT-2006-062193, COMPARE, CIT5- CT-2005-028857, and SHARELIFE, CIT4-CT-2006-028812) and through the 7th Framework Programme (SHARE-PREP, N° 211909, SHARE-LEAP, N° 227822 and SHARE M4, N° 261982). Additional funding from the U.S. National Institute on Aging (U01 AG09740-13S2, P01 AG005842, P01 AG08291, P30 AG12815, R21 AG025169, Y1-AG-4553-01, IAG BSR06-11,OGHA 04-064 and R03AG041397), the German Ministry of Education and Research and the Deutsche Forschungsgemeinschaft (e.g. BO934/10-1) as well as from various national sources is gratefully acknowledged (see www.share-project.org for a full list of funding institutions).

structure of the Stata program ("do-file") that we use to extract information from the scientific release files of SHARE to generate *easy*SHARE. We also describe some procedures in more detail, e.g. the generation of age at interview or how we deal with possible deviations in supposedly stable information. Section 6 provides an example of how to add more SHARE variables to *easy*SHARE.

2. Guidelines

The creation of longitudinal interdisciplinary databases such as the U.S. Health and Retirement Study (HRS), the English Longitudinal Study on Ageing (ELSA), and the Survey of Health, Ageing and Retirement in Europe (SHARE) has helped enormously to understand the life situation of older people in Europe and the US. One special interest of these studies is to understand the process of ageing in its multifaceted dimensions by facilitating interdisciplinary research on topics like health and health behaviour, wealth, retirement, social networks, etc. (cf. Börsch-Supan et al. 2013c, Banks et al. 2012, National Institute on Aging 2007). A main challenge of studies in these fields of research is to identify causal pathways. Two crucial ingredients for successful identification are a set of reliable and multidimensional measures at different points in time and sufficiently exogenous policy variation shaping the socio-economic environment and the repondents´ behaviour.

A major significance of SHARE lies in the second ingredient: exogenous policy variation. The strategy of SHARE and its sister surveys thus is to exploit cross-national variation. The essential argument is that the variety of circumstances and policies is in general much larger across countries than within a single country. Therefore, researchers and policy makers can learn from what has happened and what has been tried elsewhere. That also is one important reason to further increase *cross-country comparability* in *easy*SHARE by enhancing comparability with the United States, i.e. with the HRS.

To facilitate longitudinal analyses, another substantial guideline is to attain a maximum of *cross-wave comparability*. The majority of those variables and indices included in the dataset have been collected in all waves of SHARE. Of course, the third wave of data collection on respondents´ life histories (SHARELIFE) is a challenge to this principle. Due to its retrospective scope, the majority of variables in SHARELIFE does not contain the same information collected in the regular panel waves that focus on respondents´ current living conditions. Consequently, many variables of the *easy*SHARE dataset contain missing values for the third wave.

The *inclusion of new variables and indices* that are helpful for users but have not been made available in the SHARE data so far is another guideline. Among the examples are a generated age variable, a wave-specific couple identifier and a broad variety of health measures and indices, most of them adapted to the HRS.

A further guideline is to reduce the complexity of data handling and data preparation by *copying information*. Time-constant information that is only asked once in the baseline interview (i.e. height or educational level) is copied to all later records of the person.

Additionally, information collected only from one person of a couple or in a household is transferred to all respective respondents if applicable.

Furthermore, we aim to reduce complexity by providing an *extended missing code scheme*, selecting variables that have a *low share of missing information*, and by *documenting complex filtering* in target variables, if nessesary.

3. Structure of the data

Panel data like *easy*SHARE can be displayed in long or in wide format. We store *easy*SHARE in long format, i.e. the data lines represent the respondents and there is one separate line of data for each wave the respondent participated in. For example, if there are four observations with the same respondent identifier (`mergeid`), it means that this respondent took part in all four waves of SHARE. The variables or columns store the information reported. One advantage of the long format is that for most statistical packages this is the default setting for panel analyses.

Table 1 shows the structure of the data for two respondents. The first respondent is male and was interviewed in Austria, which can be derived from the prefix "AT" in the `mergeid` or from the country variables (`country` or `country_mod`). This respondent took part in all four waves as represented by one line per wave. The second respondent who was interviewed in Sweden (prefix "SE") participated in wave 1, did not participate in wave 2 and wave 3, and then again took part in wave 4. Hence, for this respondent the dataset contains only two lines. The information on respondents´ wave participation is also stored in the variable `wavepart`. You can use this variable to select balanced panels, e.g. all respondents who participated in all four waves.

Table 1: Data structure

Mergeid	wave	country	wavepart
AT-986403-01	1	Austria	1234
AT-986403-01	2	Austria	1234
AT-986403-01	3	Austria	1234
AT-986403-01	4	Austria	1234
SE-209636-01	1	Sweden	14
SE-209636-01	4	Sweden	14

Another handy feature of the *easy*SHARE data is that they contain no system-missing values. Compared to the full scientific SHARE release, for the subset of variables selected, we recode the missing values to one of the following (partially new) codes:

-3: "implausible value/suspected wrong"
-7: "not yet coded"
-9: "not applicable; filtered"
-12: "don't know/refusal"
-13: "not asked in this wave"
-14: "not asked in this country"
-15: "no information"
-16: "no drop-off (information in drop-off in this wave)"

In very rare cases we set non-system-missing values to -3: "implausible value/suspected wrong". This missing code is applied e.g. when the respondent´s body mass index (`bmi_mod`) was too low (between 0 and 12) or too high (over 100) to be realistic. A few variables are not yet coded in the data of some countries. For those the missing code -7: "not yet coded" is used. The missing value -9: "not applicable; filtered" is applied in case that a question was not asked to specific respondents due to questionnaire routing. Whenever possible, we add the reason for the filtering to the label. The -9 code is only used for filtering "within the questionnaire", but not for country or wave "filters". For every question that was not asked in all available waves of SHARE, we implement the missing code -13: "not asked in this wave". As the third wave on people's life histories (SHARELIFE) collected very different information compared to the regular waves, many variables are set to the -13 code. If a question was not asked in one of the SHARE countries, we assign the code -14 "not asked in this country".

There is only one combined missing value code -12 for "don't know" and "refusal" in *easy*SHARE. The reason is that when combining two or more variables of the full release of SHARE into a new variable, we cannot distinguish between possibly different missing codes of the original variables. For consistency within *easy*SHARE we use the combined "don't know/refusal" code for all variables. The code -15: "no information" is used for system-missing values that are not explained by any of the described reasons, for example in cases where respondents may not have completed the whole interview.

For most respondents the interview ends with the self-completion of a paper-and-pencil questionnaire – the so-called drop-off questionnaire. It includes additional questions on issues like mental and physical health, health care and social networks. If respondents did not fill in and send back the drop-off questionnaire or we cannot correctly link the latter, we use the missing code -16: "no drop-off (information in drop-off in this wave)".

4. Content of *easy*SHARE

The major criterions for including variables in *easy*SHARE are cross-country and cross-wave comparability and comparability with the HRS (see above). Moreover, we try to cover the central topics of SHARE and attend to requests by SHARE users.

We select the following topics for *easy*SHARE:

1) **Demographics**: age, gender, country of birth, citizenship, education, religion, marital status, age and gender of partner

2) **Household composition**: living with partner in the same household, household size, children living in the household

3) **Social support and network**: mother/father alive, number of children, residential proximity of children, number of grandchildren, number of living siblings, social activities, received and given social support

4) **Childhood conditions**: number of books at age ten, occupation of main breadwinner at age ten, relative mathematical skills at age ten, relative language skills at age ten

5) **Health and health behaviour:** self-perceived health, number of chronic diseases, mental health variables, depression scale EURO-D, CASP-12 index for quality of life and well-being, health care utilization, grip strength, body mass index, smoking and drinking behaviour, vigorous activities/sports

6) **Functional limitation indices**: mobility index, large muscle index, activities of daily living index, gross motor skills index, fine motor skills index, instrumental activities of daily living index, cognitive functions

7) **Work and money**: current job situation, term of main job, working hours per week, satisfaction with main job, early retirement plans, able to make ends meet, monthly expenditure on food, wave-specific household income percentiles

Some variables in *easy*SHARE are copies of the respective variables included in the main release of SHARE that were only complemented by recoding the system-missing values due to country-/wave-missing patterns and questionnaire filtering. These variables have the same name in *easy*SHARE as in the SHARE main release.

Variables that have been specifically generated for *easy*SHARE receive new variable names. Indices or scores are named according to conventions (i.e. CASP score, IADLA, etc.). If we use generated variables these are named similarly to the variable names in the generated variable modules of the SHARE main release. Variables that have been modified, but do not represent a new concept, have got an additional "_mod" suffix.

5. The Stata program generating *easy*SHARE

The Stata program, i.e. Stata "do-file", that generates the *easy*SHARE dataset based on the scientific release files of SHARE is included in full length as an appendix. For the most part, we aim for a comprehensible code that also allows users with little experience to trace the generation of the *easy*SHARE dataset. In some instances, however, we use some of the more advanced data recoding features provided by Stata. This section describes the structure of the program and gives details on the parts that we regard as requiring additional explanation.

In a first step we generate a couple of folders to store the various datasets we use to generate *easy*SHARE. We also copy into a new folder system the datasets of the regular scientific release of SHARE we use for *easy*SHARE. There are two reasons to engage in this time- and disc-space-intensive procedure. First, it is to make sure that we do not accidentally modify the original release files of SHARE. Second, it is to document which version of the release files is used to build the respective version of *easy*SHARE.

*easy*SHARE release 1.0.0 is based on the respective current release of each wave, i.e. release 2.6.0 for wave 1 and wave 2, release 1.0.0 for SHARELIFE and release 1.1.1 for wave 4. Apart from the coverscreen module (cv_r) and the drop-off questionnaire, we use information from the CAPI modules demographics (dn respectively st for SHARELIFE), activities (ac), behavioural risks (br), cognitive functioning (cf), children (ch), consumption (co), the SHARELIFE childhood section (cs), employment (ep), grip strength (gs), health care (hc), physical health (ph), social networks (sn) and social support (sp). From the generated variable modules we include gv_health, gv_isced and the imputations module.

We continue with the generation of a wave-specific couple identifier variable (`coupleid`) that is equal for both partners of a couple and is used to assign information that only one partner in a couple gave on behalf of both and to generate the partner demographic variables. The ID-like coupleid is easier to handle than the set of ego and partner identification variables supplied in the regular SHARE release files (`cvid` and `cvidp`). Note that the final *easy*SHARE dataset only stores the observations of *household members who were interviewed* in the respective wave. Hence, and in contrast to the coverscreen module that includes all household members irrespective of interview participation, there is not always a second line with the same `coupleid` because not all partners were interviewed in each wave.

Hereafter we start to extract variables from the single questionnaire module datasets of each wave. We open each of the datasets and keep only those variables that we want to include in the final *easy*SHARE dataset or that we need to generate other variables. This section is sorted per module and is the longest section of the do-file. If necessary, variables are recoded and some new variables are generated in this step before the reduced module datasets are saved separately. In the next step these reduced datasets are merged per wave so we get one dataset for each of the four waves that stores all variables extracted before. In a next step we make adjustments within the merged wave-specific datasets, like assigning information that was only gathered by one member of a household (or couple) to the other household members (respectively the partner).

Subsequently the waves are appended, thereby generating a panel 'long format dataset'. In this step we also generate a variable that stores the information on the respondents´ wave participation (`wavepart;` see table 1).

Based on the long format dataset we can check and recode several things, like deviations across waves. To generate an `age` at interview variable for as many cases as possible, we impute missing information for the date of interview and use all information on the birth year and birth month collected in the up to four interviews. This also implies to first check if the birth year and month and other stable information is consistent over time and to deal with deviating information. We impute missing information for the date of interview by substituting the wave- and country-specific mode[3]. Afterwards we check for deviations across waves for variables in which no deviations should have occured. If gender deviates between waves – which is a very rare case – we set the gender variable `female` to -3: "implausible value/suspected wrong" for all waves in which the respondent participated. We do the same for the few deviations across waves in the year of birth. If the month of birth deviates, we are less strict. Here we take the mode month of all self-reported month-of-birth values (`dn002_` in the SHARE main release data)[4]. If we have no information on the month of birth, we simply assume that these respondents are born in June. If available, we also replace missing information on gender, year and month of birth by values collected in other waves. Only then do we generate the respondents´ age and age of partner variables. Note that even though we make a couple of replacements in various variables, the resulting `age` variable can only deviate up to approx. 2 years. This is due to the fact that we never change or impute the year of birth information, and all other changes will introduce a maximum of 11 or 12 months of deviation to the true values. We regard the magnitude of these (potential) "measurement" errors in general as less harmful than a listwise deletion of all cases with the described problems. Whether or not to modify the data in the ways described is, however, mainly a decision that depends on the specific research question.

In the subsequent part of the program we transfer time-constant information collected once, usually in the first interview of a respondent – the so-called baseline interview – to later waves. This is done for `isced_r` that contains the ISCED-97[5] values, years of education (`eduyears`), country of birth (`birth_country`), the respondents´ `citizenship` and height (`ph013_` in the SHARE main release data). The latter is used to generate the body mass index (bmi). For various reasons, e.g. if the first baseline interview was not completed or interviewers mixed up the respondents, it is possible for us to have the same stable information from more than one interview. In few cases, we then observe the deviating information. As we have no way of determining which of the reports is accurate we ignore all information given.

[3] If the mode was not unique (equal number of interviews in two different months or years in the respective country) we take the ealier month or year.

[4] We again use the minimum month, if the mode was not unique. This occurs often, e.g. when we have two reports that deviate.

[5] ISCED stands for International Standard Classification of Education. SHARE assumes that its respondents who are mostly aged 50 or older do not upgrade or change their education level.

Furthermore, some time-variant information in the SHARE main release is asked and stored in ways that call for an easier handling in *easy*SHARE. If, for example, the information on marital status, living siblings and parents, or smoking behaviour did not change, the actual question is not asked in SHARE and hence the target variable is system-missing. Only a screening question that assesses whether the detail has changed since the previous interview stores this information. Therefore, we copy the information from the previous interview to the present one if the respondents indicate that the respective information has not changed. To further reduce the number of missing values in the variables `mother_alive` and `father_alive` we use information that is gathered in the social networks module of wave 4. We assume that parents who are mentioned as part of the respondents´ network are alive.

In this step we also generate the bmi variables `bmi_mod` and `bmi2_mod`. `bmi_mod` is calculated after the information on height that is only asked once in the baseline interview has been copied to subsequent waves. In `bmi2_mod` we categorise the bmi-values into underweight (bmi up to18.5), normal (bmi between 18.6 and 24.9), overweight (bmi between 25 and 29.9), and obese (bmi 30 and above). Afterwards we add, reassign, and complete labels where necessary. In the next steps the *easy*SHARE missing code scheme is completed. The final procedures include the dropping of the auxiliary variables and re-ordering of the variables. Ultimately, the dataset is compressed and saved in the formats of the two most widespread statistical packages Stata and SPSS.

The above explanations show that in the generation of the *easy*SHARE dataset we make various assumptions and decisions. For many research questions and analyses these assumptions might be adequate. Of course, there are research questions and analyses for which our procedures are not appropriate or not the best solution. All *easy*SHARE users should be aware of this.

6. How to include other variables

The *easy*SHARE dataset is restricted to specific variables chosen according to the guidelines described above. Of course, there are many other variables in the SHARE data that might be of interest for users. One solution is to modify the *easy*SHARE do-file to extract more variables. The advantage is that the routines for imputing missing information or for solving deviations in stable information can easily be used for additional variables if necessary. Users should be aware that this usually requires changes at multiple points in the do-file. Therefore, in the following we describe a second strategy to extend the *easy*SHARE dataset using the example of respondents´ area of residence that is stored in the gv_housing module. The variable contains information as to whether respondents live in a big city, suburb, large town, small town or rural area, respectively a village.

If the SHARE main release data are not downloaded yet, users first have to download the gv_housing modules of waves 1, 2 and 4. The first step of the Stata command below opens the wave-specific gv_housing modules, generates a `wave` variable and keeps the variables `areabldg`, `mergeid` and `wave`. In wave 4 the variable name is slightly different

(areabldgi) because it is not based on both the housing (ho) and interviewer (iv) module like in previous waves but only on the iv-module. Additionally, in wave 4 the information on area of residence is only provided for the household respondent. Hence, we assign the information to the other household members.

```
use  "C:\SHARE\data\sharew1_rel2-6-0_gv_housing.dta", clear
gen wave = 1
keep mergeid wave areabldg
save "C:\SHARE\data\area_w1.dta", replace


use  "C:\SHARE\data\sharew2_rel2-6-0_gv_housing.dta", clear
gen wave = 2
keep mergeid wave areabldg
save "C:\SHARE\data\area_w2.dta", replace


use  "C:\SHARE\data\sharew4_rel1-1-1_gv_housing.dta", clear
gen wave = 4
gen hhresp_help = 1 if areabldgi != .
sort hhid4 hhresp
replace areabldgi = areabldgi[_n-1] if areabldgi == . & hhid4 == hhid4[_n-1]
rename areabldgi areabldg
keep mergeid wave areabldg
save "C:\SHARE\data\area_w4.dta", replace
```

Afterwards the stored wave-specific area datasets are appended and this new dataset is saved.

```
use  "C:\SHARE\data\area_w1.dta", clear
append using "C:\SHARE\data\area_w2.dta"
append using "C:\SHARE\data\area_w4.dta"
save "C:\SHARE\data\area_appended.dta", replace
```

Then we merge the *easy*SHARE dataset and the appended area datafile using the person identifier mergeid and the wave variable.

```
use  "C:\SHARE\data\easySHARE_rel1-0-0.dta", clear
merge 1:1 mergeid wave using "C:\SHARE\data\area_appended.dta"
```

In a final step we assign the *easy*SHARE missing code scheme to the newly included variable areabldg and save the extended *easy*SHARE dataset.

```
replace areabldg = -13 if wave  == 3
replace areabldg = -12 if areabldg == -1 | areabldg == -2
replace areabldg = -15 if areabldg ==.

lab def areabldg -15 "no information"                ///
                 -13 "not asked in this wave"        ///
                 -12 "don´t know / refusal", add

numlabel areabldg, add

save "C:\SHARE\data\easySHARE_area.dta", replace
```

When integrating new variables into the *easy*SHARE dataset, users should be aware of possible deviations across waves like in the example above. Other possible deviations across waves are different question texts or response options.

7. Summary

The simplified *easy*SHARE dataset contains all respondents of the four waves of data collection but a restricted subset of variables. It is stored as a panel dataset in long format covering all SHARE countries and hence is tailor-made for teaching longitudinal as well as country-comparative analyses.

This working paper aims to contribute to a better understanding and handling of the *easy*SHARE dataset by explaining the data structure, the content as well as the guidelines that the selection of variables is based on. Additionally, we illustrated the Stata program ("do-file") that we used to generate *easy*SHARE. In the process of generating the dataset we had to make various assumptions and decisions that users should be aware of. For many analyses these assumptions might be adequate. For other research questions our procedures might be less appropriate. In the final section of this paper we showed how users can add other SHARE variables to the *easy*SHARE dataset using the example of respondents´ area of residence.

A full list of all variables included in the dataset is stored in the *easy*SHARE release guide (Hunkler et al. 2013). The release guide describes in detail all variables and all implemented modifications. Additionally, it provides an exemplary analysis using both the statistical packages Stata and SPSS. We recommend that all *easy*SHARE users study the release guide carefully before starting to work with the dataset.

References

Banks, J., J. Nazroo, and A. Steptoe (Eds.) (2012): The Dynamics of Ageing: Evidence from the English Longitudinal Study of Ageing 2002-10 (Wave 5). London: The Institute for Fiscal Studies.

Börsch-Supan, A. and H. Jürges (Eds.) (2005): The Survey of Health, Ageing and Retirement in Europe – Methodology. Mannheim: Mannheim Research Institute for the Economics of Aging (MEA).

Börsch-Supan, A., A. Brugiavini, H. Jürges, J. Mackenbach, J. Siegrist, and G. Weber (2005): Health, ageing and retirement in Europe – First results from the Survey of Health, Ageing and Retirement in Europe. Mannheim: Mannheim Research Institute for the Economics of Aging (MEA).

Börsch-Supan, A., A. Brugiavini, H. Jürges, A. Kapteyn, J. Mackenbach, J. Siegrist, and G. Weber (2008). First results from the Survey of Health, Ageing and Retirement in Europe (2004-2007). Starting the longitudinal dimension. Mannheim: Mannheim Research Institute for the Economics of Aging (MEA).

Börsch-Supan, A., M. Brandt, K. Hank, and M. Schröder (Eds.) (2011): The individual and the welfare state. Life histories in Europe. Heidelberg: Springer.

Börsch-Supan, A., M. Brandt, C. Hunkler, T. Kneip, J. Korbmacher, F. Malter, B. Schaan, S. Stuck, and S. Zuber (2013a): "Data Resource Profile: The Survey of Health, Ageing and Retirement in Europe (SHARE)", *International Journal of Epidemiology*, 42, 4: 992-1001.

Börsch-Supan, A., M. Brandt, and M. Schröder (2013b): SHARELIFE – One century of life histories in Europe, *Advances in Life Course Research*, 18, 1: 1-5.

Börsch-Supan, A., M. Brandt, H. Litwin, and G. Weber (Eds.) (2013c): Active ageing and solidarity between generations in Europe: First results from SHARE after the economic crisis. Berlin: De Gruyter.

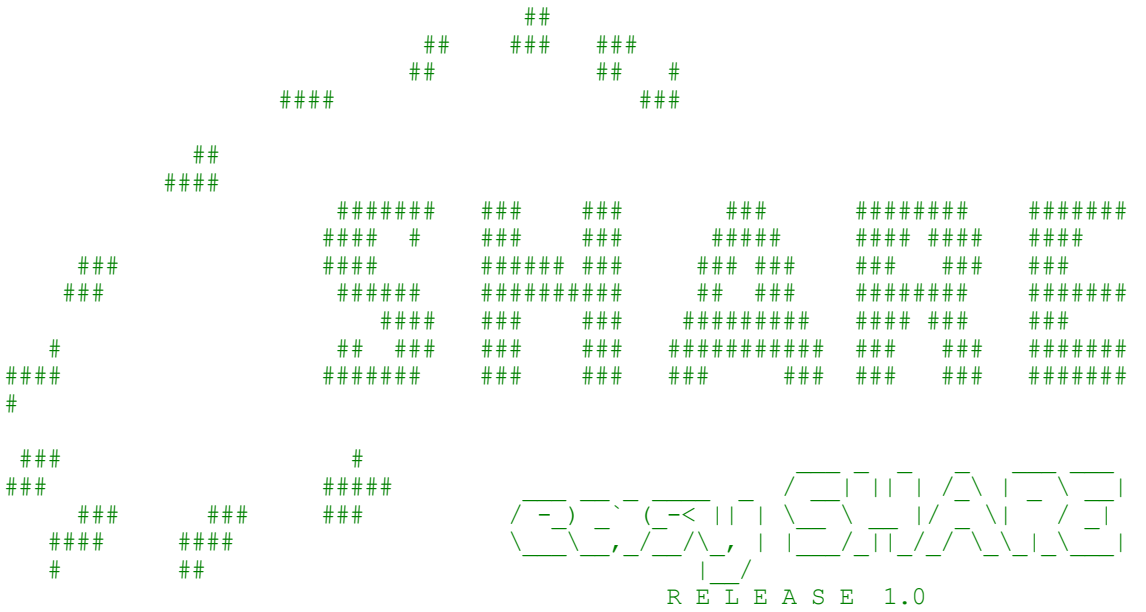
Hunkler, C., S. Gruber, A. Orban, S. Stuck, and M. Brandt (2013): Release Guide to *easy*SHARE Release 1.0.0, Munich: Munich Center for the Economics of Aging (MEA) http://www.share-project.org/data-access-documentation/easyshare.html.

Malter, F. and A. Börsch-Supan (Eds.) (2013): SHARE Wave 4: Innovations & Methodology. Munich: Munich Center for the Economics of Aging (MEA).

National Institute on Aging (2007): Growing Older in America: The Health and Retirement Study. Washington, DC: National Institutes of Health.

Schröder, M. (2011): Retrospective Data Collection in the Survey of Health, Ageing and Retirement in Europe. SHARELIFE Methodology. Mannheim: Mannheim Research Institute for the Economics of Aging (MEA).

Appendix: Stata do-file generating *easy*SHARE



- All countries & various modules of SHARE waves 1,2,3,4 -

```
Authors: Christian Hunkler, Stefan Gruber, Agnes Orban, and Stephanie Stuck.
Date:    November, 2013

This Stata program generates easySHARE based on the main release of SHARE,
it is based on the data of:

    wave 1  release 2.6.0
    wave 2  release 2.6.0
    wave 3  release 1.0.0 // SHARELIFE
    wave 4  release 1.1.1

*****************************************************************************/

*----[ O v e r v i e w  o f  C o n t e n t s ]---------------------------------
*----[ 0. Stata Version & Settings]-------------------------------------------
*----[ 1. Define paths and open log file]-------------------------------------
*----[ 2. Define w, m, c_* globals]-------------------------------------------
*----[ 3. Copy Main SHARE Release to easySHARE directory]---------------------
*----[ 4. Generate CoupleIDs based on cvid/cvidp]-----------------------------
*----[ 5. Extract & Recode Variables from cv_r]-------------------------------
*----[ 6. Extract & Recode Variables from DN / ST ]---------------------------
*----[ 7. Extract & Recode Variables from AC ]--------------------------------
*----[ 8. Extract & Recode Variables from BR ]--------------------------------
*----[ 9. Extract & Recode Variables from CF]--------------------------------
*----[10. Extract & Recode Variables from CH ]--------------------------------
*----[11. Extract & Recode Variables from CO ]--------------------------------
*----[12. Extract & Recode Variables from CS ]--------------------------------
*----[13. Extract & Recode Variables from EP ]--------------------------------
*----[14. Extract & Recode Variables from GS ]--------------------------------
*----[15. Extract & Recode Variables from HC ]--------------------------------
*----[16. Extract & Recode Variables from PH ]--------------------------------
*----[17. Extract & Recode Variables from SN ]--------------------------------
*----[18. Extract & Recode Variables from Sp ]--------------------------------
*----[19. Extract & Recode Variables from GV_Health]-------------------------
*----[20. Extract & Recode Variables from GV_ISCED ]-------------------------
*----[21. Extract & Recode Variables from DROPOFF]---------------------------
*----[22. Extract & Recode Variables from imputations]-----------------------
*----[23. Merge modules per wave ]-------------------------------------------
*----[24. Other recodes per wave ]-------------------------------------------
```

```
*----[25. Append waves to panel long format & integrate "long" variables]------
*----[26. Fix date intv., year/month birth, gender, & partnervars & gen age]---
*----[27. Transfer information collected once (in baseline interviews)]--------
*----[28. Pass on information to next wave that may have changed/not changed]--
*----[29. Fix & re-generate variables, labels, etc.]--------------------------
*----[30. Implement/complete wave/country skip patterns]----------------------
*----[31. Integrate DK/RF and implement no information missing code]-----------
*----[32. Keep, add easy missing codes & labels, order, data labels & save]----

*******************************************************************************/

*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------

*----[ 0. Stata Version & Settings]---------------------------------------------

version 12.1
clear
clear matrix
set more off

*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------

*----[ 1. Define paths and open log file]---------------------------------------

*>> Define location of waves 1,2 (rel. 2.6.0), wave 3/SHARELIFE (rel. 1.0.0),
*   and wave 4 (rel. 1.1.1)

   global wave1 "R:\DataCleaning\wave1\data\main\release_2.6.0\scrambled_final"
   global wave2 "R:\DataCleaning\wave2\data\main\release_2.6.0\scrambled_final"
   global wave3 "R:\SHARE_Docu\ReleasedData\Release_1.0.0_Sharelife"
   global wave4 "R:\SHARE_Docu\ReleasedData\Release_1.1.1_Wave_4_2013-03-28"

   global relw1 "rel2-6-0"
   global relw2 "rel2-6-0"
   global relw3 "rel1"
   global relw4 "rel1-1-1"

*>> Define location of easySHARE directory (here the new dataset is generated)

    global easy "R:\Research_projects\easySHARE\Generate_easySHARE_Release_1.0"

*>> Generate data folders within the pre-existing $easy directory:

    cd $easy                        // change to directory stored above
    capture mkdir "log"             // log  folder
    capture mkdir "data"            // data folder
    capture mkdir "data/release"    // to store a copy of the ingoing datasets
    capture mkdir "data/temp"       // to store temporary data versions

*>> Name and open a log file and store in the log directory

    capture log close
    local h =  substr("`c(current_time)'",1,2) // These commands
    local m =  substr("`c(current_time)'",4,2) // are only to
    local s =  substr("`c(current_time)'",7,2) // automatically
    local d = "`c(current_date)'"              // generate the
    local u = "`c(username)'"                  // name of the log file

    log using "$easy/log/EasySHARE_LOG_`u'__`d'_`h'-`m'-`s'.log", replace
```

```
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------

*----[ 2. Define w, m, c_* globals]---------------------------------------------

// The following globals are mostly used to copy the main release data into the
// easySHARE directory. It is not really necessary to do this from within Stata
// using the rather complex loop commands. However, this is how we do it.
// In addition, these lists provide a good overview on countries, waves &
// modules.

*>> Define what waves and modules to use

    global w "1 2 3 4"
    global m "ac as br cv_h cv_r cf ch co cs dn ep gs hc hs mh ph  dropoff do sn sp
              st xt gv_health   gv_isced gv_isco imputations "

*>> Check global lists - do never change these

    global c_w1  "at be_fr be_nl ch    de dk    es fr gr    it il    nl       se   "
    global c_w2  "at be_fr be_nl ch cz de dk    es fr gr    it il ie nl pl    se   "
    global c_w3  "at be_fr be_nl ch cz de dk    es fr gr    it       nl pl    se   "
    global c_w4  "at be_fr be_nl ch cz de dk ee es fr    hu it       nl pl pt se si"

    global m_w1 "ac as    br cf ch co    cv_h cv_r dn    ep ex ft       gs    hc hh ho
                                                        iv    mh       ph
                   sp    vi ws    gv_health gv_isced gv_isco imputations dropoff"
    global m_w2 "ac as    br cf ch co cs cv_h cv_r dn    ep ex ft       gs    hc hh ho
                                                        iv    mh   pf ph
                   sp    vi ws xt gv_health gv_isced         imputations dropoff"
    global m_w3 "ac    cs  cv_r  fq fs gs gl hc hs iv    ls    mn       rc re rp
                      st       xt"
    global m_w4 "ac as bi br cf ch co cv_r dn  ep ex ft  gs  hc hh ho iv li  mh  pf ph
                 sn sp          xt gv_health  gv_isced        imputations dropoff"


*------------------------------------------------------------------------------
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------

*----[ 3. Copy Main SHARE Release to easySHARE directory]----------------------

// Here we copy the original datasets into the easySHARE project. This is not
// very efficient in terms of storage, but we can always see which data
// versions we have used to build easySHARE, even if we accidently delete the
// data in folder we copy from.

quietly { // quitely loop starts

set more off
  noisily: di as result "[Copy main SHARE release to easySHARE directory]----"
  foreach w in $w { // wave loop starts
    noi di as result "Wave: " as txt "`w'"
  foreach m in $m { // module loop starts
    if strpos("${m_w`w'}","`m'") !=0 {   // check if module available in wave
        noisily: di as text _continue "         .. doing: mod `m', w`w'"

        copy "${wave`w'}\sharew`w'_${relw`w'}_`m'.dta" ///
             "$easy\data\release\sharew`w'_`m'.dta" , replace

        noisily: di  as text " ... DONE :-)"
    }   // end of check if module is available in wave
    else noisily: di as text   "      NOT: module `m' not in wave `w'"

  } // module loop end
  } // wave loop end
}   // quietly loop end
```

```
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------

*----[ 4. Generate CoupleIDs based on cvid/cvidp]-----------------------------

// The wave-specific couple-identifiers are used below to store information
// that only the family respondents gave, in the respective line of data of
// their partners.

// It can also be used for other purposes. However, note that while in the cv_r
// files all household members have a line and hence each coupleID is assigned
// to two lines, the final easySHARE dataset only stores the observations of
// responding household members in each wave. Hence, there will not be a
// second line with the same coupleID for each respondent (because not all
// partners were interviewed).

foreach w in $w {  // loop through all available waves to coupleID

    use $easy\data\release\sharew`w'_cv_r.dta, clear

        sort hhid`w' cvid cvidp

    *>> generate a temporary wave specific coupleID
        sort    hhid`w' mergeid cvid cvidp
        bysort  hhid`w': gen max  =_n
        bysort  hhid`w': gen temp =_n
        sum max
        global max=`r(max)'
        drop max

    *>> generate coupleID as string version
        gen coupleid`w' = hhid+"-"+string(temp)+"_"
        forvalues i=1(1)$max { // loop through all (potential) hh members

            replace coupleid`w' = coupleid`w'[_n-`i'] ///
                        if cvid   ==   cvidp[_n-`i'] ///
                         & hhid`w'== hhid`w'[_n-`i']
            replace coupleid`w' ="" if cvidp==.
            lab var coupleid`w' "couple identifier wave `w'"

        } // end of `i' loop - going through the household members


*>> checks the generated coupleid:
        assert coupleid`w' != "" if cvidp!=.
        assert coupleid`w' == "" if cvidp==.

        if "`w'"!="3" duplicates report coupleid`w' if coupleid`w' != ""
        if "`w'"=="3" duplicates report coupleid`w' if coupleid`w' != ""  & ///
                                hhid3!="AT-392101-B" & ///
                                hhid3!="AT-600652-A" & /// Within
                                hhid3!="AT-619140-B" & /// these
                                hhid3!="CZ-338422-A" & /// households
                                hhid3!="ES-452543-B" & /// we have
                                hhid3!="ES-456752-B" & /// known
                                hhid3!="ES-922367-B" & /// errors
                                hhid3!="FR-282620-B" & /// in the
                                hhid3!="GR-587441-B" & /// w3 cv_r
                                hhid3!="GR-738471-A" & ///
                                hhid3!="PL-176894-A" & ///
                                hhid3!="SE-049893-A"

        assert r(unique_value) * 2 == r(N)

    save $easy\data\temp\sharew`w'_cv_r_a.dta, replace

} // end of wave loop
```

```
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------

*----[ 5. Extract & Recode Variables from cv_r]-------------------------------

*>> w1:

use $easy\data\temp\sharew1_cv_r_a.dta, clear
    gen     wave=1
    lab var wave      "wave"
    lab var hhsize    "household size"
    lab var int_year  "interview year"
    lab var int_month "interview month"
    recode gender 2=1 1=0, gen(female)
    lab var female "gender: female=1, male=0"
    lab def female 1 "female" 0 "male"
    lab val female female
    recode mstat (1 2 = 1) (3=3), gen(partnerinhh)

    // As long as we have the non-responding partners in the file, we have to
    // store gender and age of partner of respondents, this is not possible
    // for a considerable amount of partners, after dropping the lines of those
    // with no interview.

    sort coupleid1
    gen     gender_partner=.
    replace gender_partner=female[_n-1] if coupleid1==coupleid1[_n-1] ///
                                        & coupleid1!=""
    replace gender_partner=female[_n+1] if coupleid1==coupleid1[_n+1] ///
                                        & coupleid1!=""
    replace gender_partner=-9 if partnerinhh==3 | partnerinhh==97
    lab var gender_partner "gender of partner: female=1, male=0"
    lab val gender_partner lblfemale
    lab def lblfemale       ///
            0 "0. male"     ///
            1 "1. female"   ///
           -9 "filtered: single or no partner in household", add

    sort coupleid1
    foreach var in yrbirth mobirth {
        gen     `var'_partner = .
        replace `var'_partner = `var'[_n-1]  if coupleid1==coupleid1[_n-1] ///
                                             & coupleid1!=""
        replace `var'_partner = `var'[_n+1]  if coupleid1==coupleid1[_n+1] ///
                                             & coupleid1!=""
    }

    * re-number  coupleID
    gsort -coupleid1 -temp

    keep if mergeid != "no int w.1"

    gen     temp2 = 1 if temp !=1 & coupleid1 != coupleid1[_n-1]          ///
                       & coupleid1 !=coupleid1[_n+1] & hhid1!=hhid1[_n+1] ///
                       & hhid1!=hhid1[_n-1] & coupleid1 !=""
    replace temp2 = 1 if temp !=1 & coupleid1 == coupleid1[_n-1]          ///
                       & hhid1!=hhid1[_n+1] & hhid1!=hhid1[_n-1]          ///
                       & coupleid1!=""
    replace temp2 = 1 if temp !=1 & coupleid1 == coupleid1[_n-1]          ///
                       & hhid1!=hhid1[_n+1] & coupleid1 !=""
    sort coupleid1
    by coupleid1: egen temp2_max = max(temp2) if coupleid1 !=""
    replace temp = 1 if temp2_max==1
    gen     coupleid_new = coupleid1 if temp2_max !=1
    replace coupleid_new = hhid+"-"+string(temp)+"_" if temp2_max ==1
    drop    coupleid1 temp temp2 temp2_max
    rename  coupleid_new coupleid1
```

```
    replace coupleid1 = subinstr(coupleid1, "_", "_w1",1)
    * interview of partner available
    gen     int_partner=.
    replace int_partner=-9 if  coupleid1==""
    replace int_partner=1  ///
               if (coupleid1==coupleid1[_n+1] | coupleid1==coupleid1[_n-1]) ///
                &  coupleid1!=""
    replace int_partner=5  ///
               if  coupleid1!=coupleid1[_n+1] & coupleid1!=coupleid1[_n-1]  ///
                &  coupleid1!=""
    lab var int_partner "interview of partner available"
    lab val int_partner lblinterviewp
    lab def lblinterviewp 1 "1. yes" ///
                          5 "5. no"  ///
                         -9 "filtered: single or no partner in household", add

    keep mergeid wave coupleid1 hhid1 female partnerinhh hhsize ///
         int_year int_month dumfamr dumhhr cvid gender_partner  ///
         yrbirth_partner mobirth_partner int_partner

save $easy\data\temp\sharew1_cv_r_b.dta, replace

*>> w2:

use $easy\data\temp\sharew2_cv_r_a.dta, clear
    gen wave=2
    lab var wave         "wave"
    lab var hhsize       "household size"
    lab var int_year_w2  "interview year"
    lab var int_month_w2 "interview month"
    recode gender 2=1 1=0, gen(female)
    lab var female "gender: female==1, male==0"
    lab def female 1 "female" 0 "male"
    lab val female female
    recode mstat (1 2 = 1) (3=3), gen(partnerinhh)
    replace partnerinhh = 1 if mstat2==1 & country==25
    replace partnerinhh = 3 if mstat2==3 & country==25
    rename int_month_w2 int_month
    rename int_year_w2  int_year

    sort coupleid2
    gen     gender_partner=.
    replace gender_partner=female[_n-1] if coupleid2==coupleid2[_n-1] ///
                                        & coupleid2!=""
    replace gender_partner=female[_n+1] if coupleid2==coupleid2[_n+1] ///
                                        & coupleid2!=""
    replace gender_partner=-9 if partnerinhh==3 | partnerinhh==97
    lab var gender_partner "gender of partner"
    lab val gender_partner lblfemale
    lab def lblfemale -9 "filtered: single or no partner in hh", add

    sort coupleid2
    foreach var in yrbirth mobirth {
        gen     `var'_partner = .
        replace `var'_partner = `var'[_n-1]  if coupleid2==coupleid2[_n-1] ///
                                            & coupleid2!=""
        replace `var'_partner = `var'[_n+1]  if coupleid2==coupleid2[_n+1] ///
                                             & coupleid2!=""
    }

    gsort -coupleid2 -temp
    keep if mergeid != "no int w.2"
    gen     temp2 = 1 if temp !=1 & coupleid2 != coupleid2[_n-1]          ///
                      & coupleid2 !=coupleid2[_n+1] & hhid2!=hhid2[_n+1] ///
                      & hhid2!=hhid2[_n-1] & coupleid2 !=""
    replace temp2 = 1 if temp !=1 & coupleid2 == coupleid2[_n-1] ///
                      & hhid2!=hhid2[_n+1] & hhid2!=hhid2[_n-1] ///
                      & coupleid2 !=""
```

```
    replace temp2 = 1 if temp !=1 & coupleid2 == coupleid2[_n-1] ///
                       & hhid2!=hhid2[_n+1] & coupleid2 !=""
    sort coupleid2
    by coupleid2: egen temp2_max = max(temp2) if coupleid2 !=""
    replace temp = 1 if temp2_max==1
    gen     coupleid_new = coupleid2 if temp2_max !=1
    replace coupleid_new = hhid+"-"+string(temp)+"_" if temp2_max ==1
    drop    coupleid2 temp temp2 temp2_max
    rename  coupleid_new coupleid2
    replace coupleid2 = subinstr(coupleid2, "_", "_w2",1)

    gen int_partner=.
    replace int_partner=-9 if  coupleid2==""
    replace int_partner=1  ///
            if (coupleid2==coupleid2[_n+1] | coupleid2==coupleid2[_n-1]) ///
             &  coupleid2!=""
    replace int_partner=5  ///
            if  coupleid2!=coupleid2[_n+1] & coupleid2!=coupleid2[_n-1]  ///
             &  coupleid2!=""
    lab var int_partner "interview of partner available"
    lab val int_partner lblinterviewp
    lab def lblinterviewp 1 "1. yes" ///
                          5 "5. no"  ///
                         -9 "filtered: single or no partner in household", add

    keep mergeid wave coupleid2 hhid2 female partnerinhh hhsize ///
         int_year int_month dumfamr dumhhr cvid gender_partner  ///
         yrbirth_partner mobirth_partner int_partner

save $easy\data\temp\sharew2_cv_r_b.dta, replace

*>> w3:

use $easy\data\temp\sharew3_cv_r_a.dta, clear
    gen wave=3
    lab var wave         "wave"
    lab var hhsize       "household size"
    lab var int_year_w3  "interview year"
    lab var int_month_w3 "interview month"
    recode gender 2=1 1=0, gen(female)
    lab var female "gender: female==1, male==0"
    lab def female 1 "female" 0 "male"
    lab val female female
    rename mstat partnerinhh
    rename int_month_w3 int_month
    rename int_year_w3  int_year


    sort coupleid3
    gen     gender_partner=.
    replace gender_partner=female[_n-1] if coupleid3==coupleid3[_n-1] ///
                                         & coupleid3!=""
    replace gender_partner=female[_n+1] if coupleid3==coupleid3[_n+1] ///
                                          & coupleid3!=""
    replace gender_partner=-9 if partnerinhh==3 | partnerinhh==97
    lab var gender_partner "gender of partner"
    lab val gender_partner lblfemale
    lab def lblfemale -9 "filtered: single or no partner in hh", add

    sort coupleid3
    foreach var in yrbirth mobirth {
        gen     `var'_partner = .
        replace `var'_partner = `var'[_n-1]  if coupleid3==coupleid3[_n-1] ///
                                              & coupleid3!=""
        replace `var'_partner = `var'[_n+1]  if coupleid3==coupleid3[_n+1] ///
                                              & coupleid3!=""
    }
```

```
    gsort -coupleid3 -temp
    keep if mergeid != "no int w.3"
    gen     temp2 = 1 if temp !=1 & coupleid3 != coupleid3[_n-1]            ///
                      & coupleid3 !=coupleid3[_n+1] & hhid3!=hhid3[_n+1]    ///
                      & hhid3!=hhid3[_n-1] & coupleid3 !=""
    replace temp2 = 1 if temp !=1 & coupleid3 == coupleid3[_n-1] ///
                     & hhid3!=hhid3[_n+1] & hhid3!=hhid3[_n-1] ///
                      & coupleid3 !=""
    replace temp2 = 1 if temp !=1 & coupleid3 == coupleid3[_n-1] ///
                      & hhid3!=hhid3[_n+1] & coupleid3 !=""
    sort coupleid3
    by coupleid3: egen temp2_max = max(temp2) if coupleid3 !=""
    replace temp = 1 if temp2_max==1
    gen     coupleid_new = coupleid3 if temp2_max !=1
    replace coupleid_new = hhid+"-"+string(temp)+"_" if temp2_max ==1
    drop    coupleid3 temp temp2 temp2_max
    rename  coupleid_new coupleid3
    replace coupleid3 = subinstr(coupleid3, "_", "_w3",1)

    gen int_partner=.
    replace int_partner=-9 if coupleid3==""
    replace int_partner=1  ///
            if (coupleid3==coupleid3[_n+1] | coupleid3==coupleid3[_n-1])    ///
             & coupleid3!=""
    replace int_partner=5  ///
            if coupleid3!=coupleid3[_n+1] & coupleid3!=coupleid3[_n-1]      ///
             & coupleid3!=""
    lab var int_partner "interview of partner available"
    lab val int_partner lblinterviewp
    lab def lblinterviewp 1 "1. yes" ///
                          5 "5. no"  ///
                         -9 "filtered: single or no partner in household", add

    keep if deceased == 0 // we only keep alive respondents for easySHARE,
                          // in w1 & w2 the cv_r stores only alive respondents
    keep mergeid wave coupleid3 hhid3 female partnerinhh hhsize ///
         int_year int_month cvid gender_partner                 ///
         yrbirth_partner mobirth_partner int_partner

save $easy\data\temp\sharew3_cv_r_b.dta, replace

*>> w4:

use $easy\data\temp\sharew4_cv_r_a.dta, clear
    gen wave=4
    lab var wave         "wave"
    lab var hhsize        "household size"
    lab var int_year_w4  "interview year"
    lab var int_month_w4 "interview month"
    recode gender 2=1 1=0, gen(female)
    lab var female "gender: female=1, male=0"
    lab def female 1 "female" 0 "male"
    lab val female female
    rename int_month_w4 int_month
    rename int_year_w4  int_year


    sort coupleid4
    gen     gender_partner=.
    replace gender_partner=female[_n-1] if coupleid4==coupleid4[_n-1] ///
                                        & coupleid4!=""
    replace gender_partner=female[_n+1] if coupleid4==coupleid4[_n+1] ///
                                        & coupleid4!=""
    replace gender_partner=-9 if partnerinhh==3 | partnerinhh==97
    lab var gender_partner "gender of partner: female=1, male=0"
    lab val gender_partner lblfemale
    lab def lblfemale -9 "filtered: single or no partner in hh", add
```

```
    sort coupleid4
    foreach var in yrbirth mobirth {
        gen     `var'_partner = .
        replace `var'_partner = `var'[_n-1]  if coupleid4==coupleid4[_n-1] ///
                                              & coupleid4!=""
        replace `var'_partner = `var'[_n+1]  if coupleid4==coupleid4[_n+1] ///
                                              & coupleid4!=""
    }

    gsort -coupleid4 -temp
    keep if mergeid != "no int w.4"
    gen     temp2 = 1 if temp !=1 & coupleid4 != coupleid4[_n-1]          ///
                       & coupleid4 !=coupleid4[_n+1] & hhid4!=hhid4[_n+1] ///
                      & hhid4!=hhid4[_n-1] & coupleid4 !=""
    replace temp2 = 1 if temp !=1 & coupleid4 == coupleid4[_n-1]          ///
                       & hhid4!=hhid4[_n+1] & hhid4!=hhid4[_n-1]          ///
                       & coupleid4 !=""
    replace temp2 = 1 if temp !=1 & coupleid4 == coupleid4[_n-1]          ///
                       & hhid4!=hhid4[_n+1] & coupleid4 !=""
    sort coupleid4
    by coupleid4: egen temp2_max = max(temp2) if coupleid4 !=""
    replace temp = 1 if temp2_max==1
    gen     coupleid_new = coupleid4 if temp2_max !=1
    replace coupleid_new = hhid+"-"+string(temp)+"_" if temp2_max ==1
    drop    coupleid4 temp temp2 temp2_max
    rename  coupleid_new coupleid4
    replace coupleid4 = subinstr(coupleid4, "_", "_w4",1)


    gen     int_partner=.
    replace int_partner=-9 if coupleid4==""
    replace int_partner=1                                                 ///
            if (coupleid4==coupleid4[_n+1] | coupleid4==coupleid4[_n-1])  ///
             & coupleid4!=""
    replace int_partner=5                                                 ///
            if coupleid4!=coupleid4[_n+1] & coupleid4!=coupleid4[_n-1]    ///
             & coupleid4!=""
    lab var int_partner "interview of partner available"
    lab val int_partner lblinterviewp
    lab def lblinterviewp 1 "1. yes"  ///
                          5 "5. no"   ///
                         -9 "filtered: single or no partner in household", add

    keep if deceased == 0 // we only keep alive respondents for easySHARE,
                          // in w1 & w2 the cv_r stores only alive respondents
    keep    mergeid wave waveid coupleid4 hhid4 female ///
            partnerinhh hhsize int_year int_month dumfamr dumhhr cvid  ///
            gender_partner yrbirth_partner mobirth_partner int_partner

save $easy\data\temp\sharew4_cv_r_b.dta, replace


*------------------------------------------------------------------------------
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------

*----[ 6. Extract & Recode Variables from DN / ST ]----------------------------

*>> w1:

use $easy\data\release\sharew1_dn.dta, clear

  * country_mod - codes adapted to ISO-codes
    gen     country_mod =   .
    replace country_mod =  40 if country==11
    replace country_mod = 276 if country==12
    replace country_mod = 752 if country==13
```

```
    replace country_mod = 528 if country==14
    replace country_mod = 724 if country==15
    replace country_mod = 380 if country==16
    replace country_mod = 250 if country==17
    replace country_mod = 208 if country==18
    replace country_mod = 300 if country==19
    replace country_mod = 756 if country==20
    replace country_mod =  56 if country==23
    replace country_mod = 376 if country==25

    lab def country_mod  40 "Austria"       276 "Germany"   752 "Sweden"    ///
                        528 "Netherlands"   724 "Spain"     380 "Italy"     ///
                        250 "France"        208 "Denmark"   300 "Greece"    ///
                        756 "Switzerland"    56 "Belgium"   376 "Israel"    ///
                        203 "Czechia"       616 "Poland"    372 "Ireland"   ///
                        348 "Hungary"       620 "Portugal"  705 "Slovenia"  ///
                        233 "Estonia"
    lab val country_mod country_mod
    lab var country_mod "country identifier (ISO coded)"

  * birth_country - filled for born in country and codes adapted to ISO-codes
    gen        birth_country = dn005c    if dn004==5
    replace birth_country = country_mod if dn004==1
    lab var  birth_country "country of birth (ISO coded)"
    // value labels will be taken from wave 4

  * Citizenship - filled for country citizenship and codes adapted to ISO-codes
    gen        citizenship = dn008c            if dn007==5
    replace citizenship = country_mod if dn007==1
    lab var  citizenship "citizenship of respondent (ISO coded)"
    // value labels will be taken from wave 4

  * Siblings_alive
    gen        siblings_alive = dn037_ + dn036_ if dn036_>=0 & dn037>=0
    replace siblings_alive = -9 if dn034_==5
    lab var  siblings_alive ///
            "number of siblings alive (based on: dn036_,dn037_,dn034_)"
    lab def siblings_alive -9 "filtered: no siblings ever", add
    lab val siblings_alive siblings_alive

  keep  mergeid country country_mod language dn002_ dn003_ birth_country ///
        citizenship dn014_ siblings_alive dn026_1 dn026_2

save $easy\data\temp\sharew1_dn_a.dta, replace


*>> w2:

use $easy\data\release\sharew2_dn.dta, clear

  * country_mod - codes adapted to ISO-codes
    gen     country_mod =   .
    replace country_mod =  40 if country==11
    replace country_mod = 276 if country==12
    replace country_mod = 752 if country==13
    replace country_mod = 528 if country==14
    replace country_mod = 724 if country==15
    replace country_mod = 380 if country==16
    replace country_mod = 250 if country==17
    replace country_mod = 208 if country==18
    replace country_mod = 300 if country==19
    replace country_mod = 756 if country==20
    replace country_mod =  56 if country==23
    replace country_mod = 203 if country==28
    replace country_mod = 616 if country==29
    replace country_mod = 372 if country==30
    replace country_mod = 376 if country==25
```

```
    lab def country_mod  40 "Austria"       276 "Germany"   752 "Sweden"    ///
                        528 "Netherlands"   724 "Spain"     380 "Italy"     ///
                        250 "France"        208 "Denmark"   300 "Greece"    ///
                        756 "Switzerland"    56 "Belgium"   376 "Israel"    ///
                        203 "Czechia"       616 "Poland"    372 "Ireland"   ///
                        348 "Hungary"       620 "Portugal"  705 "Slovenia"  ///
                        233 "Estonia"       376 "Israel"
    lab val country_mod country_mod
    lab var country_mod "country identifier (ISO coded)"

  * birth_country - filled for born in country and codes adapted to ISO-codes
    gen     birth_country = dn005c   if dn004==5
    replace birth_country = country_mod if dn004==1
    lab var birth_country "country of birth (ISO coded)"
    // value labels will be taken from wave 4

  * Citizenship - filled for country citizenship and codes adapted to ISO-codes
    gen     citizenship = dn008c   if dn007==5
    replace citizenship = country_mod if dn007==1
    lab var citizenship "citizenship of respondent (ISO coded)"
    // value labels will be taken from wave 4

  * Siblings
    gen     siblings_alive = dn037_ + dn036_ if dn036_>=0 & dn037>=0
    replace siblings_alive =-9 if dn034_==5
    lab var siblings_alive ///
            "number of siblings alive (based on: dn036_, dn037_, dn034_)"
    lab def siblings_alive -9 "filtered: no siblings ever", add
    lab val siblings_alive siblings_alive

  * Years of education:
    rename dn041_ eduyears

    keep mergeid country country_mod language dn002_ dn003_ birth_country ///
         citizenship dn014_ siblings_alive dn026_1 dn026_2 dn044_ dn043_ eduyears

save $easy\data\temp\sharew2_dn_a.dta, replace

*>> w3:

use $easy\data\release\sharew3_st.dta, clear
    rename sl_st006_ dn002_
    rename sl_st007_ dn003_

  * country_mod - codes adapted to ISO-codes
    gen     country_mod =   .
    replace country_mod =  40 if country==11
    replace country_mod = 276 if country==12
    replace country_mod = 752 if country==13
    replace country_mod = 528 if country==14
    replace country_mod = 724 if country==15
    replace country_mod = 380 if country==16
    replace country_mod = 250 if country==17
    replace country_mod = 208 if country==18
    replace country_mod = 300 if country==19
    replace country_mod = 756 if country==20
    replace country_mod =  56 if country==23
    replace country_mod = 203 if country==28
    replace country_mod = 616 if country==29
    replace country_mod = 372 if country==30

    lab def country_mod  40 "Austria"       276 "Germany"   752 "Sweden"    ///
                        528 "Netherlands"   724 "Spain"     380 "Italy"     ///
                        250 "France"        208 "Denmark"   300 "Greece"    ///
                        756 "Switzerland"    56 "Belgium"   376 "Israel"    ///
                        203 "Czechia"       616 "Poland"    372 "Ireland"   ///
                        348 "Hungary"       620 "Portugal"  705 "Slovenia"  ///
                        233 "Estonia"
```

```
    lab val country_mod country_mod
    lab var country_mod "country identifier (ISO coded)"

    keep mergeid hhid3 country country_mod language dn002_ dn003_ sl_st011_
           // sl_st011_ taken to replace  missing lines/info in cv_r file

save $easy\data\temp\sharew3_st_a.dta, replace

*>> w4:

use $easy\data\release\sharew4_dn.dta, clear

  * country_mod - codes adapted to ISO-codes
    gen     country_mod =   .
    replace country_mod =  40 if country==11
    replace country_mod = 276 if country==12
    replace country_mod = 752 if country==13
    replace country_mod = 528 if country==14
    replace country_mod = 724 if country==15
    replace country_mod = 380 if country==16
    replace country_mod = 250 if country==17
    replace country_mod = 208 if country==18
    replace country_mod = 300 if country==19
    replace country_mod = 756 if country==20
    replace country_mod =  56 if country==23
    replace country_mod = 203 if country==28
    replace country_mod = 616 if country==29
    replace country_mod = 348 if country==32
    replace country_mod = 620 if country==33
    replace country_mod = 705 if country==34
    replace country_mod = 233 if country==35

    lab def country_mod  40 "Austria"       276 "Germany"   752 "Sweden"   ///
                        528 "Netherlands"   724 "Spain"     380 "Italy"    ///
                        250 "France"        208 "Denmark"   300 "Greece"   ///
                        756 "Switzerland"    56 "Belgium"   376 "Israel"   ///
                        203 "Czechia"       616 "Poland"    372 "Ireland"  ///
                        348 "Hungary"       620 "Portugal"  705 "Slovenia" ///
                        233 "Estonia"
    lab val country_mod country_mod
    lab var country_mod "country identifier (ISO coded)"

  * birth_country - filled for born in country and codes adapted to ISO-codes
    gen     birth_country = dn005c    if dn004==5
    replace birth_country = country_mod if dn004==1
    lab var birth_country "country of birth (ISO coded)"

  * Citizenship - filled for country citizenship and codes adapted to ISO-codes
    gen     citizenship   = dn008c            if dn007==5
    replace citizenship = country_mod if dn007==1
    lab var citizenship "citizenship of respondent (ISO coded)"

  * Siblings
    gen     siblings_alive= dn037_ + dn036_ if dn036_>=0 & dn037>=0
    replace siblings_alive=-9 if dn034_==5
    lab var siblings_alive ///
            "number of siblings alive (based on: dn036_, dn037_, dn034_)"
    lab def siblings_alive -9 "filtered: no siblings ever", add
    lab val siblings_alive siblings_alive

    rename dn041_ eduyears

    keep mergeid country country_mod language dn002_ dn003_ birth_country ///
         citizenship dn014_ siblings_alive dn026_1 dn026_2 dn044_ dn043_  ///
         eduyears dn005c dn008c
         // dn005c dn008c are kept for their value labels

save $easy\data\temp\sharew4_dn_a.dta, replace
```

```
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------

*----[ 7. Extract & Recode Variables from AC ]--------------------------------

*>> w1:

use $easy\data\release\sharew1_ac.dta, clear

    keep mergeid ac002d1 ac002d2 ac002d3 ac002d4 ac002d5 ac002d6 ac002d7 ///
         ac002dno

save $easy\data\temp\sharew1_ac_a.dta, replace

*>> w2:

use $easy\data\release\sharew2_ac.dta, clear

    mvdecode ac02* ac03* ac01*, mv(-1=.a \ -2=.b)

  * CESD Score; Recode feelings dummies for scale construction: 1=yes and 0=no
    recode ac027_ 5=0
    recode ac028_ 5=0
    recode ac029_ 5=0
    recode ac031_ 5=0
    recode ac033_ 5=0
    recode ac034_ 5=0
    recode ac030_ 5=0
    recode ac032_ 5=0

    gen cesd2 = ac027_ + ac028_ + ac029_ + ac031_ + ac033_ + ac034_ + ///
                    (1-ac030_)+(1-ac032_)
    lab var cesd2 "CESD score wave 2 (high is depressed)"

  * Quality of Life Score (CASP-12)
    // inverse into  *_inv (high value / high control etc.)
    recode ac017_ (4=1) (3=2) (2=3) (1=4) (else=.) , gen(ac017_inv)
    recode ac020_ (4=1) (3=2) (2=3) (1=4) (else=.) , gen(ac020_inv)
    recode ac021_ (4=1) (3=2) (2=3) (1=4) (else=.) , gen(ac021_inv)
    recode ac022_ (4=1) (3=2) (2=3) (1=4) (else=.) , gen(ac022_inv)
    recode ac023_ (4=1) (3=2) (2=3) (1=4) (else=.) , gen(ac023_inv)
    recode ac024_ (4=1) (3=2) (2=3) (1=4) (else=.) , gen(ac024_inv)
    recode ac025_ (4=1) (3=2) (2=3) (1=4) (else=.) , gen(ac025_inv)

    // Subscale Control
    gen     con= ac014_ + ac015_ + ac016_
    alpha        ac014_   ac015_   ac016_

    // Subscale Autonomy
    gen     aut= ac017_inv + ac018_ + ac019_
    alpha        ac017_inv   ac018_   ac019_

    // Subscale Pleasure
    gen     ple= ac020_inv + ac021_inv + ac022_inv
    alpha        ac020_inv   ac021_inv   ac022_inv

    // Subscale Self-Realisation
    gen     sel= ac023_inv + ac024_inv + ac025_inv
    alpha        ac023_inv   ac024_inv   ac025_inv

    // CASP
    gen     casp= con + aut + ple + sel
    lab var casp "CASP quality of life (high is high quality)"
    mvencode ac02* ac03* ac01*, mv(.a=-1 \ .b=-2)
    keep mergeid ac002d1 ac002d2 ac002d3 ac002d4 ac002d5 ac002d6 ac002d7 ///
         ac002dno cesd2 casp

save $easy\data\temp\sharew2_ac_a.dta, replace
```

```
*>> w4:

use $easy\data\release\sharew4_ac.dta, clear
    mvdecode ac02* ac03* ac01*, mv(-1=.a \ -2=.b)

  * Quality of Life Score (CASP-12)

    // inverse into  *_inv (high value / high control etc.)
    recode ac017_ (4=1) (3=2) (2=3) (1=4) (else=.) , gen(ac017_inv)
    recode ac020_ (4=1) (3=2) (2=3) (1=4) (else=.) , gen(ac020_inv)
    recode ac021_ (4=1) (3=2) (2=3) (1=4) (else=.) , gen(ac021_inv)
    recode ac022_ (4=1) (3=2) (2=3) (1=4) (else=.) , gen(ac022_inv)
    recode ac023_ (4=1) (3=2) (2=3) (1=4) (else=.) , gen(ac023_inv)
    recode ac024_ (4=1) (3=2) (2=3) (1=4) (else=.) , gen(ac024_inv)
    recode ac025_ (4=1) (3=2) (2=3) (1=4) (else=.) , gen(ac025_inv)

    // Subscale Control
    gen     con = ac014_ + ac015_ + ac016_
    alpha         ac014_   ac015_   ac016_

    // Subscale Autonomy
    gen aut= ac017_inv + ac018_ + ac019_
    alpha    ac017_inv   ac018_   ac019_
    // Subscale Pleasure
    gen     ple= ac020_inv + ac021_inv + ac022_inv
    alpha        ac020_inv   ac021_inv   ac022_inv

    // Subscale Self-Realisation
    gen     sel = ac023_inv + ac024_inv + ac025_inv
    alpha         ac023_inv   ac024_inv   ac025_inv

    gen     casp = con + aut + ple + sel
    lab var casp "CASP quality of life (high is high quality)"

    mvencode ac02* ac03* ac01*, mv(.a=-1 \ .b=-2)

    keep mergeid casp

save $easy\data\temp\sharew4_ac_a.dta, replace

*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------

*----[ 8. Extract & Recode Variables from BR ]---------------------------------

*>> w1:

use $easy\data\release\sharew1_br.dta, clear

    recode br010_ (7=1) (6=2) (5=3) (4=4) (3=5) (2=6) (1=7) (-1=-1) (-2=-2) ///
                  (else=.)
    lab def lblalcohol       -2 "refusal" -1 "don't know" 1 "not at all"   ///
                2 "less than once a month"  3 "once or twice a month"      ///
                4 "once or twice a week"    5 "three or four days a week"  ///
                6 "five or six days a week" 7 "almost every day"
    lab val br010_ lblalcohol
    rename br010_ br010_mod

    keep mergeid br001_ br002_ br010_mod br015_

save $easy\data\temp\sharew1_br_a.dta, replace
```

```
*>> w2:

use $easy\data\release\sharew2_br.dta, clear

    recode br010_ (7=1) (6=2) (5=3) (4=4) (3=5) (2=6) (1=7) (-1=-1) (-2=-2) ///
                  (else=.)
    lab def lblalcohol -2 "refusal" -1 "don't know" 1 "not at all"      ///
                2 "less than once a month" 3 "once or twice a month"    ///
                4 "once or twice a week" 5 "three or four days a week"  ///
                6 "five or six days a week" 7 "almost every day"
    lab val br010_ lblalcohol
    rename br010_ br010_mod

    keep mergeid br001_ br002_ br010_mod br015_

save $easy\data\temp\sharew2_br_a.dta, replace


*>> w4:

use $easy\data\release\sharew4_br.dta, clear

    recode br010_ (7=1) (6=2) (5=3) (4=4) (3=5) (2=6) (1=7) (-1=-1) (-2=-2) ///
                  (else=.)
    lab def lblalcohol -2 "refusal" -1 "don't know" 1 "not at all"      ///
                2 "less than once a month" 3 "once or twice a month"    ///
                4 "once or twice a week" 5 "three or four days a week"  ///
                6 "five or six days a week" 7 "almost every day"
    lab val br010_ lblalcohol
    rename br010_ br010_mod

    keep mergeid br001_ br002_ br010_mod br015_

save $easy\data\temp\sharew4_br_a.dta, replace

*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------

*----[ 9. Extract & Recode Variables from CF]---------------------------------

*>> w1:

use $easy\data\release\sharew1_cf.dta, clear
    gen recall_1=cf008tot
    gen recall_2=cf016tot
    lab var recall_1 "recall of words, first trial (based on cf008tot)"
    lab var recall_2 "recall of words, delayed (based on cf016tot)"
    keep mergeid recall*
save $easy\data\temp\sharew1_cf_a.dta, replace

*>> w2:

use $easy\data\release\sharew2_cf.dta, clear
    gen recall_1=cf008tot
    gen recall_2=cf016tot
    lab var recall_1 "recall of words, first trial (based on cf008tot)"
    lab var recall_2 "recall of words, delayed (based on cf016tot)"
    keep mergeid recall*
save $easy\data\temp\sharew2_cf_a.dta, replace

*>> w4: recall in wave 4 is generated based on gv_health-module

use $easy\data\release\sharew4_cf.dta, clear
    keep mergeid
save $easy\data\temp\sharew4_cf_a.dta, replace
```

```
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------

*----[10. Extract & Recode Variables from CH ]--------------------------------

*>> w1:

use $easy\data\release\sharew1_ch.dta, clear

    gen ch007_hh=0
    replace ch007_hh=1 if ch007_1==1 | ch007_1==2 | ch007_2 ==1 | ch007_2==2 ///
        | ch007_3 ==1 | ch007_3 ==2 | ch007_4 ==1 | ch007_4 ==2              ///
        | ch007_5 ==1 | ch007_5 ==2 | ch007_6 ==1 | ch007_6 ==2 | ch007_7==1 ///
        | ch007_7 ==2 | ch007_8 ==1 | ch007_8 ==2 | ch007_9 ==1 | ch007_9==2 ///
        | ch007_10==1 | ch007_10==2 | ch007_11==1 | ch007_11==2              ///
        | ch007_12==1 | ch007_12==2 | ch007_13==1 | ch007_13==2              ///
        | ch007_14==1 | ch007_14==2                                          ///
        | ch007_15==1 | ch007_15==2 | ch007_16==1 | ch007_16==2

    replace ch007_hh =. if ch001_  ==. | ch001_==-1 | ch001_==-2
    replace ch007_hh =. if ch007_1 ==. & ch007_2==. & ch007_3==. &            ///
            ch007_4  ==. & ch007_5 ==. & ch007_6==. & ch007_7==. & ch007_8==. ///
          & ch007_9  ==. & ch007_10==. & ch007_11==. & ch007_12==.            ///
          & ch007_13 ==. & ch007_14==. & ch007_15==. & ch007_16==.
    replace ch007_hh=-9 if ch001_==0

    lab var ch007_hh ///
     "at least one child lives in household/building (based on ch007_1-ch007_16)"
    lab def lblch007_hh -9 "filtered: no children" 0 "no" 1 "yes"
    lab val ch007_hh lblch007_hh

    gen ch007_km=0
    replace ch007_km=1 if ch007_1==3 | ch007_2==3 | ch007_3==3 | ch007_4==3  ///
        | ch007_5==3 | ch007_6==3                                           ///
        | ch007_7==3 | ch007_8==3 | ch007_9==3 | ch007_10==3 | ch007_11==3   ///
        | ch007_12==3 | ch007_13==3                                          ///
        | ch007_14==3 | ch007_15==3 | ch007_16==3
    replace ch007_km=. if ch001_==. | ch001_==-1 | ch001_==-2
    replace ch007_km=. if ch007_1==. & ch007_2==. & ch007_3==. & ch007_4==.  ///
        & ch007_5==. & ch007_6==. & ch007_7==. & ch007_8==.                  ///
        & ch007_9==. & ch007_10==. & ch007_11==. & ch007_12==. & ch007_13==. ///
        & ch007_14==. & ch007_15==. & ch007_16==.
    replace ch007_km=-9 if ch001_==0
    lab var ch007_km ///
     "at least one child lives less than 1km away (based on ch007_1-ch007_16)"
    lab def lblch007_km -9 "filtered: no children" 0 "no" 1 "yes"
    lab val ch007_km lblch007_km

    replace ch021_ = -9 if ch001_ == 0
    lab def lblch021_ -9 "filtered: no children" ///
                          -2 "refusal" -1 "don´t know"
    lab val ch021_ lblch021_
    rename ch021_ ch021_mod

    keep mergeid ch001_ ch021_mod ch023_ ch007_hh ch007_km
save $easy\data\temp\sharew1_ch_a.dta, replace

*>> w2:

use $easy\data\release\sharew2_ch.dta, clear
    gen ch007_hh=0
    replace ch007_hh=1 if ch007_1==1 | ch007_1==2 | ch007_2==1 | ch007_2==2 ///
        | ch007_3==1 | ch007_3==2 | ch007_4==1 | ch007_4==2                  ///
        | ch007_5==1 | ch007_5==2 | ch007_6==1 | ch007_6==2 | ch007_7==1     ///
        | ch007_7==2 | ch007_8==1 | ch007_8==2 | ch007_9==1 | ch007_9==2     ///
        | ch007_10==1 | ch007_10==2 | ch007_11==1 | ch007_11==2             ///
        | ch007_12==1 | ch007_12==2 | ch007_13==1 | ch007_13==2             ///
```

```
        | ch007_14==1 | ch007_14==2                                          ///
        | ch007_15==1 | ch007_15==2 | ch007_16==1 | ch007_16==2
    replace ch007_hh=. if ch001_==. | ch001_==-1 | ch001_==-2
    replace ch007_hh=. if ch007_1==. & ch007_2==. & ch007_3==.               ///
        & ch007_4==. & ch007_5==. & ch007_6==. & ch007_7==. & ch007_8==.     ///
        & ch007_9==. & ch007_10==. & ch007_11==. & ch007_12==. & ch007_13==. ///
        & ch007_14==. & ch007_15==. & ch007_16==.
    replace ch007_hh=-9 if ch001_==0
    lab var ch007_hh ///
     "at least one child lives in household/building (based on ch007_1-ch007_16)"
    lab def lblch007_hh -9 "filtered: no children" 0 "no" 1 "yes"
    lab val ch007_hh lblch007_hh

    gen ch007_km=0
    replace ch007_km=1 if ch007_1==3 | ch007_2==3 | ch007_3==3       ///
        | ch007_4==3 | ch007_5==3 | ch007_6==3                       ///
        | ch007_7==3 | ch007_8==3 | ch007_9==3 | ch007_10==3         ///
        | ch007_11==3 | ch007_12==3 | ch007_13==3                   ///
        | ch007_14==3 | ch007_15==3 | ch007_16==3
    replace ch007_km=. if ch001_==. | ch001_==-1 | ch001_==-2
    replace ch007_km=. if ch007_1==. & ch007_2==. & ch007_3==.       ///
    & ch007_4==. & ch007_5==. & ch007_6==. & ch007_7==. & ch007_8==. ///
        & ch007_9==. & ch007_10==. & ch007_11==. & ch007_12==.       ///
        & ch007_13==. & ch007_14==. & ch007_15==. & ch007_16==.
    replace ch007_km=-9 if ch001_==0
    lab var ch007_km ///
     "at least one child lives less than 1km away (based on ch007_1-ch007_16)"
    lab def lblch007_km -9 "filtered: no children" 0 "no" 1 "yes"
    lab val ch007_km lblch007_km

    replace ch021_ = -9 if ch001_ == 0
    lab def lblch021_ -9 "filtered: no children" ///
                          -2 "refusal" -1 "don´t know"
    lab val ch021_ lblch021_
    rename ch021_ ch021_mod

    keep mergeid ch001_ ch021_mod ch023_ ch007_hh ch007_km
save $easy\data\temp\sharew2_ch_a.dta, replace

*>> w4:

use $easy\data\release\sharew4_ch.dta, clear

    gen ch007_hh=0
    replace ch007_hh=1 if ch007_1==1 | ch007_1==2 | ch007_2==1           ///
        | ch007_2==2 | ch007_3==1 | ch007_3==2 | ch007_4==1 | ch007_4==2 ///
        | ch007_5==1 | ch007_5==2 | ch007_6==1 | ch007_6==2 | ch007_7==1 ///
        | ch007_7==2 | ch007_8==1 | ch007_8==2 | ch007_9==1 | ch007_9==2 ///
        | ch007_10==1 | ch007_10==2 | ch007_11==1 | ch007_11==2          ///
        | ch007_12==1 | ch007_12==2 | ch007_13==1 | ch007_13==2          ///
        | ch007_14==1 | ch007_14==2                                       ///
        | ch007_15==1 | ch007_15==2 | ch007_16==1 | ch007_16==2
    replace ch007_hh=. if ch001_==. | ch001_==-1 | ch001_==-2
    replace ch007_hh=. if ch007_1==. & ch007_2==. & ch007_3==.           ///
        & ch007_4==. & ch007_5==. & ch007_6==. & ch007_7==. & ch007_8==. ///
        & ch007_9==. & ch007_10==. & ch007_11==. & ch007_12==.           ///
        & ch007_13==. & ch007_14==. & ch007_15==. & ch007_16==.
    replace ch007_hh=-9 if ch001_==0
    lab var ch007_hh ///
     "at least one child lives in household/building (based on ch007_1-ch007_16)"
    lab def lblch007_hh -9 "filtered: no children" 0 "no" 1 "yes"
    lab val ch007_hh lblch007_hh

    gen ch007_km=0
    replace ch007_km=1 if ch007_1==3 | ch007_2==3 | ch007_3==3           ///
        | ch007_4==3 | ch007_5==3 | ch007_6==3                           ///
        | ch007_7==3 | ch007_8==3 | ch007_9==3 | ch007_10==3             ///
        | ch007_11==3 | ch007_12==3 | ch007_13==3                       ///
```

```
        | ch007_14==3 | ch007_15==3 | ch007_16==3
    replace ch007_km=. if ch001_==. | ch001_==-1 | ch001_==-2
    replace ch007_km=. if ch007_1==. & ch007_2==. & ch007_3==. & ch007_4==. ///
        & ch007_5==. & ch007_6==. & ch007_7==. & ch007_8==.                 ///
        & ch007_9==. & ch007_10==. & ch007_11==. & ch007_12==.              ///
        & ch007_13==. & ch007_14==. & ch007_15==. & ch007_16==.
    replace ch007_km=-9 if ch001_==0
    lab var ch007_km ///
     "at least one child lives less than 1km away (based on ch007_1-ch007_16)"
    lab def lblch007_km -9 "filtered: no children" 0 "no" 1 "yes"
    lab val ch007_km lblch007_km

    replace ch021_ = -9 if ch001_ == 0
    lab def lblch021_ -9 "filtered: no children" ///
                      -2 "refusal" -1 "don´t know"
    lab val ch021_ lblch021_
    rename ch021_ ch021_mod

    keep mergeid ch001_ ch021_mod ch023_ ch007_hh ch007_km ch524_ ch526*

save $easy\data\temp\sharew4_ch_a.dta, replace

*------------------------------------------------------------------------------
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------

*----[11. Extract & Recode Variables from CO ]---------------------------------

*>> w1:

use  $easy\data\release\sharew1_co.dta, clear
     keep mergeid co007_
save $easy\data\temp\sharew1_co_a.dta, replace

*>> w2:

use  $easy\data\release\sharew2_co.dta, clear
     keep mergeid co007_
save $easy\data\temp\sharew2_co_a.dta, replace

*>> w4:

use  $easy\data\release\sharew4_co.dta, clear
     keep mergeid co007_
save $easy\data\temp\sharew4_co_a.dta, replace

*------------------------------------------------------------------------------
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------

*----[12. Extract & Recode Variables from CS ]---------------------------------

*>> w3:

use $easy\data\release\sharew3_cs.dta, clear

    recode sl_cs010_ 9=-9
    lab def lblmaths -2 "refusal" -1 "don't know"                    ///
                     -9 "filtered: did not go to school"             ///
                      1 "much better" 2 "better" 3 "about the same"  ///
                      4 "worse" 5 "much worse"
    lab val sl_cs010_ lblmaths
    rename sl_cs010_ sl_cs010_mod

    recode sl_cs010a_ .=-9 if sl_cs010a_==-9
    lab def lbllanguage -2 "refusal" -1 "don't know"                  ///
                        -9 "filtered: did not go to school"           ///
                         1 "much better" 2 "better" 3 "about the same" ///
```

```
                        4 "worse" 5 "much worse"
    lab val sl_cs010a_ lbllanguage
    rename sl_cs010a_ sl_cs010a_mod

    keep mergeid sl_cs008_ sl_cs009_ sl_cs010_mod sl_cs010a_mod

save $easy\data\temp\sharew3_cs_a.dta, replace

*------------------------------------------------------------------------------
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------

*----[13. Extract & Recode Variables from EP ]---------------------------------

*>> w1:

use $easy\data\release\sharew1_ep.dta, clear

    lab def lblep005_ -2 "refusal" -1 "don't know"      1 "retired"    ///
                       2 "employed or self-employed"    3 "unemployed" ///
                       4 "permanently sick or disabled" 5 "homemaker"  ///
                      97 "other"
    lab val ep005_ lblep005_

    replace ep009_1=-9 if ep005_!=2 & ep005_>0 & ep005_ !=.
    lab def lblep009_1 -9 "filtered: not worked" -2 "refusal"          ///
                       -1 "don't know"            1 "employee"         ///
                        2 "civil servant"         3 "self-employed"
    lab val ep009_1 lblep009_1
    rename ep009_1 ep009_

    replace ep011_1=-9 if ep009_==3 |  ep009_==-9
        // -9 for "not worked" added
    lab def lblep011_1 -9 "filtered: not worked or self-employed"   ///
                       -2 "refusal" -1 "don't know" 1 "short-term"  ///
                        2 "permanent"
    lab val ep011_1 lblep011_1
    rename ep011_1 ep011_mod

    replace ep013_1=-9 if ep002_==5 & ep003_==5 & ep005_!=2 & ep005_>0
    lab def lblep013_1 -9 "filtered: not worked" -2 "refusal" ///
                       -1 "don't know"
    lab val ep013_1 lblep013_1
    rename ep013_1 ep013_mod

    replace ep026_=-9 if ep003_==5 & ep005_!=2 & ep005_>0
    lab def lblep026_ -9 "filtered: not worked" -2 "refusal"        ///
                      -1 "don't know"            1 "strongly agree" ///
                       2 "agree"                 3 "disagree"       ///
                       4 "strongly disagree"
    lab val ep026_ lblep026_
    rename ep026_ ep026_mod

    replace ep036_=-9 if ep005_ !=. & (ep005_>2 | ep005_==1)
    lab def lblep036_ -9 "filtered: not worked" -2 "refusal" ///
                      -1 "don't know" 1 "yes" 5 "no"
    lab val ep036_ lblep036_
    rename ep036_ ep036_mod

    keep mergeid ep005_ ep009_ ep011_mod ep013_mod ep026_mod ep036_mod

save $easy\data\temp\sharew1_ep_a.dta, replace
```

```
*>> w2:

use $easy\data\release\sharew2_ep.dta, clear

    lab def lblep005_ -2 "refusal" -1 "don't know"      1 "retired"     ///
                       2 "employed or self-employed"    3 "unemployed"  ///
                       4 "permanently sick or disabled" 5 "homemaker"   ///
                      97 "other"
    lab val ep005_ lblep005_

    replace ep009_=-9 if ep005_!=2 & ep005_>0 & ep005_ !=.
    lab def lblep009_ -9 "filtered: not worked" -2 "refusal"            ///
                      -1 "don't know"            1 "employee"           ///
                       2 "civil servant"         3 "self-employed"
    lab val ep009_ lblep009_

    replace ep011_=-9 if ep009_==3 | ep009_==-9
    lab def lblep011_ -9 "filtered: not worked or self-employed"        ///
                      -2 "refusal" -1 "don't know" 1 "short-term"       ///
                       2 "permanent"
    lab val ep011_ lblep011_
    rename ep011_ ep011_mod

    replace ep013_=-9 if ep005_!=2 & ep005_>0 & ep005_ !=.
    lab def lblep013_ -9 "filtered: not worked" -2 "refusal"    ///
                      -1 "don't know"
    lab val ep013_ lblep013_
    rename ep013_ ep013_mod

    replace ep026_=-9 if ep005_!=2 & ep005_>0 & ep005_ !=.
    lab def lblep026_ -9 "filtered: not worked" -2 "refusal"              ///
                      -1 "don't know"     1 "strongly agree"                ///
                       2 "agree"          3 "disagree" 4 "strongly disagree"
    lab val ep026_ lblep026_
    rename ep026_ ep026_mod

    replace ep036_=-9 if ep005_ !=. & (ep005_>2 | ep005_==1)
    lab def lblep036_ -9 "filtered: not worked" -2 "refusal" ///
                      -1 "don't know"  1 "yes" 5 "no"
    lab val ep036_ lblep036_
    rename ep036_ ep036_mod

    keep mergeid ep005_ ep009_ ep011_mod ep013_mod ep026_mod ep036_mod

save $easy\data\temp\sharew2_ep_a.dta, replace

*>> w4:

use $easy\data\release\sharew4_ep.dta, clear

    lab def lblep005_ -2 "refusal" -1 "don't know"       1 "retired"     ///
                       2 "employed or self-employed"     3 "unemployed"  ///
                       4 "permanently sick or disabled"  5 "homemaker"   ///
                      97 "other"
    lab val ep005_ lblep005_

    replace ep009_=-9 if ep005_!=2 & ep005_>0 & ep005_ !=.
    lab def lblep009_ -9 "filtered: not worked" -2 "refusal" ///
                      -1 "don't know"        1 "employee"        ///
                       2 "civil servant"     3 "self-employed"
    lab val ep009_ lblep009_

    replace ep011_=-9 if ep009_==3 | ep009_==-9
    lab def lblep011_ -9 "filtered: not worked or self-employed"  ///
                      -2 "refusal" -1 "don't know" 1 "short-term" ///
                       2 "permanent"
    lab val ep011_ lblep011_
    rename ep011_ ep011_mod
```

```
    replace ep013_=-9 if ep005_!=2 & ep005_>0 & ep005_ !=.
    lab def lblep013_ -9 "filtered: not worked" -2 "refusal" ///
                      -1 "don't know"
    lab val ep013_ lblep013_
    rename ep013_ ep013_mod

    replace ep026_=-9 if ep005_!=2 & ep005_>0 & ep005_ !=.
    lab def lblep026_ -9 "filtered: not worked" -2 "refusal"        ///
                      -1 "don't know" 1 "strongly agree"            ///
                       2 "agree"  3 "disagree" 4 "strongly disagree"
    lab val ep026_ lblep026_
    rename ep026_ ep026_mod

    replace ep036_=-9 if ep005_ !=. & (ep005_>2 | ep005_==1)
    lab def lblep036_ -9 "filtered: not worked" -2 "refusal" ///
                      -1 "don't know" 1 "yes" 5 "no"
    lab val ep036_ lblep036_
    rename ep036_ ep036_mod

    keep mergeid ep005_ ep009_ ep011_mod ep013_mod ep026_mod ep036_mod

save $easy\data\temp\sharew4_ep_a.dta, replace

*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------

*----[14. Extract & Recode Variables from GS ]--------------------------------

// In wave 3/SHARELIFE grip strength has been measured, but the main
// release of SHARE does not yet include a generated health module
// providing the maxgrip variable. Therefore we generate it here for wave 3.
// For the other waves we use the variable included in the gv_health module.

use $easy\data\release\sharew3_gs.dta, clear

*>> recode left and right vars
    generate left1 = sl_gs006_
    replace  left1 = .  if   (sl_gs006_ <= 0 | sl_gs006_ >= 100)

    generate left2 = sl_gs007_
    replace  left2 = .  if   (sl_gs007_ <= 0 | sl_gs007_ >= 100)

    generate right1 = sl_gs008_
    replace  right1 = .  if  (sl_gs008_ <= 0 | sl_gs008_ >= 100)

    generate right2 = sl_gs009_
    replace  right2 = .  if  (sl_gs009_ <= 0 | sl_gs009_ >= 100)


    // Counting no. of right hand grip strength measures
    egen  nright = robs(right1 right2)

    // Counting no. of left hand grip strength measures
    egen  nleft = robs(left1 left2)

*>> generating maxgrip:
    // if difference between the two measurements on one hand is < 20
    // only if two valid measurements on the measuring hand

    generate goodR = 1  if  (nright == 2 &  abs(right1 - right2) < 20)
    generate goodL = 1  if  (nleft  == 2 &  abs(left1  - left2 ) < 20)

    egen    maxgrip = rmax(left1 left2 right1 right2) if (goodR==1 & goodL==1)

    replace maxgrip = max(left1, left2)    if  (goodR == . & goodL == 1)
    replace maxgrip = max(right1, right2)  if  (goodR == 1 & goodL == .)
```

```
    lab var maxgrip  "max. of grip strength measures"

    keep mergeid maxgrip

save $easy\data\temp\sharew3_gs_a.dta, replace

*------------------------------------------------------------------------------
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------

*----[15. Extract & Recode Variables from HC ]--------------------------------

*>> w1:

use $easy\data\release\sharew1_hc.dta, clear

    gen hc032c=.
    replace hc032c=1   if hc032d1==1 | hc032d2==1 | hc032d3==1
    replace hc032c=0   if hc032dno==1
    replace hc032c=-1  if hc032d1==-1 & hc032d2==-1 & hc032d3==-1
    replace hc032c=-2  if hc032d1==-2 & hc032d2==-2 & hc032d3==-2
    replace hc032c=-9  if hc029==3
    replace hc032c=-14 if country==19
    lab var hc032c ///
     "utilization of home care last 12 months (based on hc032*, hc029)"
    lab def lblhc032c  -9 "filtered: permanently in nursing home"  ///
                      -2 "refusal" -1 "don't know" 0 "no" 1 "yes"
    lab val hc032c lblhc032c

    replace hc038_=-14 if country==11 | country==12
    lab def yesno -14 "not asked in this country", add
    rename hc038_ hc038_mod

    keep mergeid hc002_ hc012_ hc029_ hc032c hc038_mod

save $easy\data\temp\sharew1_hc_a.dta, replace

*>> w2:

use $easy\data\release\sharew2_hc.dta, clear

    gen hc032c=.
    replace hc032c=1 if hc032d1==1 | hc032d2==1 | hc032d3==1
    replace hc032c=0 if hc032dno==1
    replace hc032c=-9 if hc029==3
    replace hc032c=-2 if hc032d1==-2 & hc032d2==-2 & hc032d3==-2
    replace hc032c=-1 if hc032d1==-1 & hc032d2==-1 & hc032d3==-1
    replace hc032c=-14 if country ==29
    lab var hc032c ///
     "utilization of home care last 12 months (based on hc032*, hc029)"
    lab def lblhc032c -14 "not asked in this country"               ///
                      -9 "filtered: permanently in nursing home"   ///
                      -2 "refusal" -1 "don't know" 0 "no" 1 "yes"
    lab val hc032c lblhc032c

    replace hc038_=-14 if country==11 | country==12
    lab def yesno -14 "not asked in this country", add
    rename hc038_ hc038_mod

    keep mergeid hc002_ hc012_ hc029_ hc032c hc038_mod

save $easy\data\temp\sharew2_hc_a.dta, replace
```

```
*>> w4: hc032c and hc038 not included in w4

use $easy\data\release\sharew4_hc.dta, clear

    gen hc032c=-13
    lab var hc032c ///
     "utilization of home care last 12 months (based on hc032*, hc029)"
    lab def lblhc032c -13 "not asked in this wave"
    lab val hc032c lblhc032c

    gen hc038_=-13
    lab var hc038_ "received care from private providers"
    lab def lblhc038_ -13 "not asked in this wave"
    lab val hc038_ lblhc038_
    rename hc038_ hc038_mod

    keep mergeid hc002_ hc012_ hc029_ hc032c hc038_mod

save $easy\data\temp\sharew4_hc_a.dta, replace

*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------

*----[16. Extract & Recode Variables from PH ]---------------------------------

*>> w1:

use $easy\data\release\sharew1_ph.dta, clear

    mvdecode ph048* ph049* ph006d? ph006d?? , mv(-1=.a \ -2=.b)

    gen     mobilityind = ph048d1 + ph049d2 + ph048d4 + ph048d5
    lab var mobilityind "mobility index (high: has difficulties)"

    gen     lgmuscle = ph048d2 + ph048d3 + ph048d6 + ph048d8
    lab var lgmuscle "large muscle index (high: has difficulties)"

    gen     adlwa = ph049d1 + ph049d3 + ph049d4
    lab var adlwa ///
     "activities of daily living w&h index (high: has difficulties)"

    gen     adla = ph049d1 + ph049d3 + ph049d4 + ph049d2 + ph049d5
    lab var adla "activities of daily living index (high: has difficulties)"

    gen     grossmotor = ph048d1 + ph049d2 + ph048d5 + ph049d3
    lab var grossmotor "gross motor skills index (high: has difficulties)"

    gen     finemotor = ph048d10 + ph049d4 + ph049d1
    lab var finemotor "fine motor skills index (high: has difficulties)"

    gen     iadla = ph049d10 + ph049d11 + ph049d13
    lab var iadla ///
     "instrumental activities of daily living index (high: has difficulties)"

    gen     iadlza= ph049d10 + ph049d11 + ph049d13 + ph049d9 + ph049d8
    lab var iadlza ///
     "instrumental activities of daily living index 2 (high: has difficulties)"

    egen    chronic_mod = rowtotal(ph006d1  ph006d2  ph006d3  ph006d4  ///
                                   ph006d5  ph006d6  ph006d8  ph006d10 ///
                                   ph006d11 ph006d12 ph006d13 ph006d14)
    lab var chronic_mod "number of chronic diseases (easySHARE version)"

    mvencode ph048* ph049* ph006d? ph006d??, mv(.a=-1 \ .b=-2)

    replace mobilityind = -12 if ph048d1==-1 | ph048d1==-2 ///
                               | ph049d2==-1 | ph049d2==-2
```

```
    replace lgmuscle    = -12 if ph048d1==-1 | ph048d1==-2
    replace adlwa       = -12 if ph049d1==-1 | ph049d1==-2
    replace adla        = -12 if ph049d1==-1 | ph049d1==-2
    replace grossmotor  = -12 if ph048d1==-1 | ph048d1==-2 ///
                               | ph049d2==-1 | ph049d2==-2
    replace finemotor   = -12 if ph048d1==-1 | ph048d1==-2 ///
                               | ph049d2==-1 | ph049d2==-2
    replace iadla       = -12 if ph049d1==-1 | ph049d1==-2
    replace iadlza      = -12 if ph049d1==-1 | ph049d1==-2
    replace chronic_mod = -12 if ph006d1==-1 | ph006d1==-2
    replace chronic_mod =   . if ph006d1==  .

    keep mergeid mobilityind lgmuscle adlwa adla grossmotor finemotor iadla ///
         iadlza ph012 ph013 chronic_mod

save $easy\data\temp\sharew1_ph_a.dta, replace

*>> w2:

use $easy\data\release\sharew2_ph.dta, clear
    mvdecode ph048* ph049* ph006d? ph006d??, mv(-1=.a \ -2=.b)

    gen     mobilityind = ph048d1 + ph049d2 + ph048d4 + ph048d5
    lab var mobilityind "mobility Index (high: has difficulties)"

    gen lgmuscle = ph048d2 + ph048d3 + ph048d6 + ph048d8
    lab var lgmuscle "large Muscle Index (high: has difficulties)"

    gen     adlwa = ph049d1 + ph049d3 + ph049d4
    lab var adlwa ///
     "actvities of Daily Living W&H Index (high: has difficulties)"

    gen     adla= ph049d1 + ph049d3 + ph049d4 + ph049d2 + ph049d5
    lab var adla "actvities of Daily Living Index (high: has difficulties)"

    gen     grossmotor = ph048d1 + ph049d2 + ph048d5 + ph049d3
    lab var grossmotor "gross Motor Skills Index (high: has difficulties)"

    gen     finemotor = ph048d10 + ph049d4 + ph049d1
    lab var finemotor "fine Motor Skills Index (high: has difficulties)"

    gen     iadla = ph049d10 + ph049d11 + ph049d13
    lab var iadla ///
     "instrumental activities of Daily Living Index (high: has difficulties)"

    gen     iadlza = ph049d10 + ph049d11 + ph049d13 + ph049d9 + ph049d8
    lab var iadlza ///
     "instrumental activities of Daily Living Index 2 (high: has difficulties)"

    egen    chronic_mod = rowtotal(ph006d1  ph006d2  ph006d3  ph006d4  ///
                                   ph006d5  ph006d6  ph006d8  ph006d10 ///
                                   ph006d11 ph006d12 ph006d13 ph006d14)
    lab var chronic_mod "number of chronic diseases (easySHARE version)"

    mvencode ph048* ph049* ph006d? ph006d??, mv(.a=-1 \ .b=-2)

    replace mobilityind = -12 if ph048d1==-1 | ph048d1==-2 ///
                               | ph049d2==-1 | ph049d2==-2
    replace lgmuscle    = -12 if ph048d1==-1 | ph048d1==-2
    replace adlwa       = -12 if ph049d1==-1 | ph049d1==-2
    replace adla        = -12 if ph049d1==-1 | ph049d1==-2
    replace grossmotor  = -12 if ph048d1==-1 | ph048d1==-2 ///
                               | ph049d2==-1 | ph049d2==-2
    replace finemotor   = -12 if ph048d1==-1 | ph048d1==-2 ///
                               | ph049d2==-1 | ph049d2==-2
    replace iadla       = -12 if ph049d1==-1 | ph049d1==-2
    replace iadlza      = -12 if ph049d1==-1 | ph049d1==-2
    replace chronic_mod = -12 if ph006d1==-1 | ph006d1==-2
```

```
    replace chronic_mod =   . if ph006d1==  .  // egen ignores sysmis

    keep mergeid mobilityind lgmuscle adlwa adla grossmotor finemotor iadla ///
         iadlza ph012 ph013 chronic_mod

save $easy\data\temp\sharew2_ph_a.dta, replace

*>> w4:

use $easy\data\release\sharew4_ph.dta, clear

    mvdecode ph048* ph049*, mv(-1=.a \ -2=.b)

    gen     mobilityind = ph048d1 + ph049d2 + ph048d4 + ph048d5
    lab var mobilityind "mobility Index (high: has difficulties)"

    gen     lgmuscle = ph048d2 + ph048d3 + ph048d6 + ph048d8
    lab var lgmuscle "large Muscle Index (high: has difficulties)"

    gen     adlwa = ph049d1 + ph049d3 + ph049d4
    lab var adlwa ///
     "activities of Daily Living W&H Index (high: has difficulties)"

    gen     adla = ph049d1 + ph049d3 + ph049d4 + ph049d2 + ph049d5
    lab var adla "activities of Daily Living Index (high: has difficulties)"

    gen     grossmotor = ph048d1 + ph049d2 + ph048d5 + ph049d3
    lab var grossmotor "gross Motor Skills Index (high: has difficulties)"

    gen     finemotor = ph048d10 + ph049d4 + ph049d1
    lab var finemotor "fine Motor Skills Index (high: has difficulties)"

    gen     iadla= ph049d10 + ph049d11 + ph049d13
    lab var iadla ///
     "instrumental acivities of Daily Living Index (high: has difficulties)"

    gen     iadlza= ph049d10 + ph049d11 + ph049d13 + ph049d9 + ph049d8
    lab var iadlza ///
     "instrumental activities of Daily Living Index 2 (high: has difficulties)"


    egen    chronic_mod = rowtotal(ph006d1  ph006d2  ph006d3  ph006d4  ///
                                   ph006d5  ph006d6  ph006d8  ph006d10 ///
                                   ph006d11 ph006d12 ph006d13 ph006d14)
    lab var chronic_mod "number of chronic diseases (easySHARE version)"

    mvencode ph048* ph049* ph048* ph049*, mv(.a=-1 \ .b=-2)

    replace mobilityind = -12 if ph048d1==-1 | ph048d1==-2 ///
                               | ph049d2==-1 | ph049d2==-2
    replace lgmuscle   = -12 if ph048d1==-1 | ph048d1==-2
    replace adlwa      = -12 if ph049d1==-1 | ph049d1==-2
    replace adla        = -12 if ph049d1==-1 | ph049d1==-2
    replace grossmotor  = -12 if ph048d1==-1 | ph048d1==-2 ///
                               | ph049d2==-1 | ph049d2==-2
    replace finemotor   = -12 if ph048d1==-1 | ph048d1==-2 ///
                               | ph049d2==-1 | ph049d2==-2
    replace iadla       = -12 if ph049d1==-1 | ph049d1==-2
    replace iadlza      = -12 if ph049d1==-1 | ph049d1==-2
    replace chronic_mod = -12 if ph006d1==-1 | ph006d1==-2
    replace chronic_mod =   . if ph006d1==  .  // egen ignores sysmis

    keep mergeid mobilityind lgmuscle adlwa adla grossmotor finemotor iadla ///
         iadlza ph012 ph013 chronic_mod

save $easy\data\temp\sharew4_ph_a.dta, replace
```

```
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------

*----[17. Extract & Recode Variables from SN]---------------------------------

*>> w4:

use $easy\data\release\sharew4_sn.dta, clear
    keep mergeid sn005_*
save $easy\data\temp\sharew4_sn_a.dta, replace

*------------------------------------------------------------------------------
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------

*----[18. Extract & Recode Variables from SP]---------------------------------

*>> w1:

use $easy\data\release\sharew1_sp.dta, clear
    keep mergeid sp002_ sp003_1 sp003_2 sp003_3 sp008_ sp009_1 sp009_2 sp009_3
save $easy\data\temp\sharew1_sp_a.dta, replace

*>> w2:

use $easy\data\release\sharew2_sp.dta, clear
    keep mergeid sp002_ sp003_1 sp003_2 sp003_3 sp008_ sp009_1 sp009_2 sp009_3
save $easy\data\temp\sharew2_sp_a.dta, replace

*>> w4:

use $easy\data\release\sharew4_sp.dta, clear

   foreach var in sp003_1sp sp003_2sp sp003_3sp sp009_1sp sp009_2sp sp009_3sp {
       replace `var' = -7 // different value labels in wave 4
   }

   rename sp003_1sp sp003_1
   rename sp003_2sp sp003_2
   rename sp003_3sp sp003_3
   rename sp009_1sp sp009_1
   rename sp009_2sp sp009_2
   rename sp009_3sp sp009_3

   keep mergeid sp002_ sp003_1 sp003_2 sp003_3 sp008_ sp009_1 sp009_2 sp009_3

save $easy\data\temp\sharew4_sp_a.dta, replace


*------------------------------------------------------------------------------
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------

*----[19. Extract & Recode Variables from GV_Health]--------------------------

*>> w1:

use $easy\data\release\sharew1_gv_health.dta, clear

    keep mergeid sphus euro* orienti numeracy maxgrip

save $easy\data\temp\sharew1_gv_health_a.dta, replace


*>> w2:

use $easy\data\release\sharew2_gv_health.dta, clear
```

```
    keep mergeid sphus euro* orienti numeracy maxgrip

save $easy\data\temp\sharew2_gv_health_a.dta, replace

*>> w3:

use $easy\data\release\sharew3_hs.dta, clear

    rename sl_ph003_ sphus
    keep mergeid     sphus

save $easy\data\temp\sharew3_hs_a.dta, replace

*>> w4:

use $easy\data\release\sharew4_gv_health.dta, clear

    gen recall_1=cf008tot
    gen recall_2=cf016tot
    lab var recall_1 "recall of words, first trial (based on cf008tot)"
    lab var recall_2 "recall of words, delayed (based on cf016tot)"
    keep mergeid sphus euro* maxgrip orienti numeracy recall*

save $easy\data\temp\sharew4_gv_health_a.dta, replace

*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------

*----[20. Extract & Recode Variables from GV_ISCED ]--------------------------

// We do not use the eduyear variable asked in wave 1, because it is coded
// based on the ISCED classification and hence something very different
// in comparison to the self-reported years of education collected in waves
// 2 and 4.

*>> w1:

use $easy\data\release\sharew1_gv_isced.dta, clear
    keep mergeid isced_r
save $easy\data\temp\sharew1_gv_isced_a.dta, replace


*>> w2: -> replace missings after appending the waves

use $easy\data\release\sharew2_gv_isced.dta, clear
    keep mergeid isced_r
save $easy\data\temp\sharew2_gv_isced_a.dta, replace


*>> w4: -> replace missings after appending the waves

use $easy\data\release\sharew4_gv_isced.dta, clear
    keep mergeid isced_r
save $easy\data\temp\sharew4_gv_isced_a.dta, replace
```

```
*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------

*----[21. Extract & Recode Variables from DROPOFF]-----------------------------

*>> w1:

use $easy\data\release\sharew1_dropoff.dta, clear
    mvdecode q4* q2*, mv(-5=.c)

  * CESD1 Score
    gen     q4_a_inv=.
    replace q4_a_inv=1 if q4_a==4
    replace q4_a_inv=2 if q4_a==3
    replace q4_a_inv=3 if q4_a==2
    replace q4_a_inv=4 if q4_a==1

    gen     q4_b_inv=.
    replace q4_b_inv=1 if q4_b==4
    replace q4_b_inv=2 if q4_b==3
    replace q4_b_inv=3 if q4_b==2
    replace q4_b_inv=4 if q4_b==1

    gen     q4_c_inv=.
    replace q4_c_inv=1 if q4_c==4
    replace q4_c_inv=2 if q4_c==3
    replace q4_c_inv=3 if q4_c==2
    replace q4_c_inv=4 if q4_c==1

    gen     q4_e_inv=.
    replace q4_e_inv=1 if q4_e==4
    replace q4_e_inv=2 if q4_e==3
    replace q4_e_inv=3 if q4_e==2
    replace q4_e_inv=4 if q4_e==1

    gen     q4_h_inv=.
    replace q4_h_inv=1 if q4_h==4
    replace q4_h_inv=2 if q4_h==3
    replace q4_h_inv=3 if q4_h==2
    replace q4_h_inv=4 if q4_h==1

    gen     q4_j_inv=.
    replace q4_j_inv=1 if q4_j==4
    replace q4_j_inv=2 if q4_j==3
    replace q4_j_inv=3 if q4_j==2
    replace q4_j_inv=4 if q4_j==1

    gen     cesd1 = (q4_a_inv + q4_b_inv + q4_c_inv + q4_e_inv + q4_h_inv ///
                             + q4_j_inv + q4_d     + q4_g                 )
    lab var cesd1 "CESD scale wave 1 (high is depressed)"

 * Quality of Life Score(CASP-12)

    gen     q2_d_inv=.
    replace q2_d_inv=1 if q2_d==4
    replace q2_d_inv=2 if q2_d==3
    replace q2_d_inv=3 if q2_d==2
    replace q2_d_inv=4 if q2_d==1

    gen     q2_g_inv=.
    replace q2_g_inv=1 if q2_g==4
    replace q2_g_inv=2 if q2_g==3
    replace q2_g_inv=3 if q2_g==2
    replace q2_g_inv=4 if q2_g==1

    gen     q2_h_inv=.
    replace q2_h_inv=1 if q2_h==4
```

```
    replace q2_h_inv=2 if q2_h==3
    replace q2_h_inv=3 if q2_h==2
    replace q2_h_inv=4 if q2_h==1

    gen     q2_i_inv=.
    replace q2_i_inv=1 if q2_i==4
    replace q2_i_inv=2 if q2_i==3
    replace q2_i_inv=3 if q2_i==2
    replace q2_i_inv=4 if q2_i==1

    gen     q2_j_inv=.
    replace q2_j_inv=1 if q2_j==4
    replace q2_j_inv=2 if q2_j==3
    replace q2_j_inv=3 if q2_j==2
    replace q2_j_inv=4 if q2_j==1

    gen     q2_k_inv=.
    replace q2_k_inv=1 if q2_k==4
    replace q2_k_inv=2 if q2_k==3
    replace q2_k_inv=3 if q2_k==2
    replace q2_k_inv=4 if q2_k==1

    gen     q2_l_inv=.
    replace q2_l_inv=1 if q2_l==4
    replace q2_l_inv=2 if q2_l==3
    replace q2_l_inv=3 if q2_l==2
    replace q2_l_inv=4 if q2_l==1

    // Subscale Control
    gen con= q2_a + q2_b + q2_c
    alpha    q2_a   q2_b   q2_c

    // Subscale Autonomy
    gen aut= q2_d_inv + q2_e + q2_f
    alpha    q2_d_inv q2_e q2_f

    // Subscale Pleasure
    gen ple= q2_g_inv + q2_h_inv + q2_i_inv
    alpha    q2_g_inv   q2_h_inv   q2_i_inv

    // Subscale Self-Realisation
    gen sel= q2_j_inv + q2_k_inv + q2_l_inv
    alpha    q2_j_inv   q2_k_inv   q2_l_inv

    gen     casp = con + aut + ple + sel
    lab var casp  "CASP quality of life (high is high quality)"

    mvencode q4* q2*, mv(.c=-5)

    keep mergeid cesd1 casp q34_re

save $easy\data\temp\sharew1_dropoff_a.dta, replace

*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------

*----[22. Extract & Recode Variables from imputations]-------------------------

*>> w1:

use $easy\data\release\sharew1_imputations.dta, clear
    drop if mergeid=="no int w.1" // imputations for partners without interview
    keep if implicat==1

  * Converting nominal euros into real amounts in prices obtaining in
  * Germany in 2005
```

```
  * fahcv: expenditure on food at home last 12 months
    gen        fahcv_mod=fahcv/pppx2004 if int_year==2004
    replace fahcv_mod=fahcv/pppx2005 if int_year==2005
    lab var  fahcv_mod "hhd monthly expenditure on food at home"
    lab val   fahcv_mod fahcv_mod

  * fohcv: expenditure on food outside the home last 12 months
    gen        fohcv_mod=fohcv/pppx2004 if int_year==2004
    replace fohcv_mod=fohcv/pppx2005 if int_year==2005
    lab var  fohcv_mod "hhd monthly expenditure on food outside home"
    lab val   fohcv_mod fohcv_mod

  * Income

  * hgtincv: household total gross income in 2003
    // in w1 all values going into hgtincv were asked for 2003
    // even for interview in 2005; this changed for w2 onwards
    gen hgtincv_p=hgtincv/pppx2003

  * country-specific income percentiles
    foreach c in 11 12 13 14 15 16 17 18 19 20 23 {
     pctile  p10_`c' = hgtincv_p if country==`c', nq(10)
     return list
     gen     income_pct_w1_`c'=.
     replace income_pct_w1_`c'=1 if hgtincv_p<=r(r1)                    ///
                                  & country==`c'
     replace income_pct_w1_`c'=2 if hgtincv_p<=r(r2) & hgtincv_p>r(r1) ///
                                  & country==`c'
     replace income_pct_w1_`c'=3 if hgtincv_p<=r(r3) & hgtincv_p>r(r2) ///
                                  & country==`c'
     replace income_pct_w1_`c'=4 if hgtincv_p<=r(r4) & hgtincv_p>r(r3) ///
                                  & country==`c'
     replace income_pct_w1_`c'=5 if hgtincv_p<=r(r5) & hgtincv_p>r(r4) ///
                                  & country==`c'
     replace income_pct_w1_`c'=6 if hgtincv_p<=r(r6) & hgtincv_p>r(r5) ///
                                  & country==`c'
     replace income_pct_w1_`c'=7 if hgtincv_p<=r(r7) & hgtincv_p>r(r6) ///
                                  & country==`c'
     replace income_pct_w1_`c'=8 if hgtincv_p<=r(r8) & hgtincv_p>r(r7) ///
                                  & country==`c'
     replace income_pct_w1_`c'=9 if hgtincv_p<=r(r9) & hgtincv_p>r(r8) ///
                                  & country==`c'
     replace income_pct_w1_`c'=10 if hgtincv_p>r(r9) & hgtincv_p<.     ///
                                  & country==`c'
     lab val income_pct_w1_`c' income_pct`c'
    }

    *one variable including all countries
    gen income_pct_w1 = .
    foreach c in 11 12 13 14 15 16 17 18 19 20 23 {
        replace income_pct_w1 =  1  if  income_pct_w1_`c'== 1
        replace income_pct_w1 =  2  if  income_pct_w1_`c'== 2
        replace income_pct_w1 =  3  if  income_pct_w1_`c'== 3
        replace income_pct_w1 =  4  if  income_pct_w1_`c'== 4
        replace income_pct_w1 =  5  if  income_pct_w1_`c'== 5
        replace income_pct_w1 =  6  if  income_pct_w1_`c'== 6
        replace income_pct_w1 =  7  if  income_pct_w1_`c'== 7
        replace income_pct_w1 =  8  if  income_pct_w1_`c'== 8
        replace income_pct_w1 =  9  if  income_pct_w1_`c'== 9
        replace income_pct_w1 = 10  if  income_pct_w1_`c'==10
    }

    lab var income_pct_w1 "hhd income percentiles wave 1"
    lab val income_pct_w1 income_pct_w1

    keep mergeid fahcv_mod fohcv_mod income_pct_w1

save $easy\data\temp\sharew1_imputations_a.dta, replace
```

```
*>> w2:

use $easy\data\release\sharew2_imputations.dta, clear
    drop if mergeid =="no int w.2"
    keep if implicat==1

  * Converting nominal euros into real amounts in prices obtaining ///
  * in Germany in 2005

  * fahcv: expenditure on food at home last 12 months
    gen     fahcv_mod=fahcv/pppx2006 if int_year==2006
    replace fahcv_mod=fahcv/pppx2007 if int_year==2007
    replace fahcv_mod=fahcv/pppx2008 if int_year==2008
    replace fahcv_mod=fahcv/pppx2009 if int_year==2009
    replace fahcv_mod=fahcv/pppx2010 if int_year==2010
    lab var fahcv_mod "hhd monthly expenditure on food at home"
    lab val fahcv_mod fahcv_mod

  * fohcv: expenditure on food outside the home last 12 months
    gen     fohcv_mod=fohcv/pppx2006 if int_year==2006
    replace fohcv_mod=fohcv/pppx2007 if int_year==2007
    replace fohcv_mod=fohcv/pppx2008 if int_year==2008
    replace fohcv_mod=fohcv/pppx2009 if int_year==2009
    replace fohcv_mod=fohcv/pppx2010 if int_year==2010
    lab var fohcv_mod "hhd monthly expenditure on food outside home"
    lab val fohcv_mod fohcv_mod

  * Income

  * hgtincv: household total net income (previous year)
    gen     hgtincv_p=hgtincv/pppx2005 if int_year==2006
    replace hgtincv_p=hgtincv/pppx2006 if int_year==2007
    replace hgtincv_p=hgtincv/pppx2007 if int_year==2008
    replace hgtincv_p=hgtincv/pppx2008 if int_year==2009
    replace hgtincv_p=hgtincv/pppx2009 if int_year==2010

  * country-specific income percentiles
    foreach c in 11 12 13 14 15 16 17 18 19 20 23 25 28 29 {
        pctile  p10_`c' = hgtincv_p if country==`c', nq(10)
        return list
        gen     income_pct_w2_`c'=.
        replace income_pct_w2_`c'= 1 if hgtincv_p<=r(r1)                    ///
                                      & country==`c'
        replace income_pct_w2_`c'= 2 if hgtincv_p<=r(r2) & hgtincv_p>r(r1) ///
                                      & country==`c'
        replace income_pct_w2_`c'= 3 if hgtincv_p<=r(r3) & hgtincv_p>r(r2) ///
                                      & country==`c'
        replace income_pct_w2_`c'= 4 if hgtincv_p<=r(r4) & hgtincv_p>r(r3) ///
                                      & country==`c'
        replace income_pct_w2_`c'= 5 if hgtincv_p<=r(r5) & hgtincv_p>r(r4) ///
                                      & country==`c'
        replace income_pct_w2_`c'= 6 if hgtincv_p<=r(r6) & hgtincv_p>r(r5) ///
                                      & country==`c'
        replace income_pct_w2_`c'= 7 if hgtincv_p<=r(r7) & hgtincv_p>r(r6) ///
                                      & country==`c'
        replace income_pct_w2_`c'= 8 if hgtincv_p<=r(r8) & hgtincv_p>r(r7) ///
                                      & country==`c'
        replace income_pct_w2_`c'= 9 if hgtincv_p<=r(r9) & hgtincv_p>r(r8) ///
                                      & country==`c'
        replace income_pct_w2_`c'=10 if hgtincv_p> r(r9) & hgtincv_p<.     ///
                                      & country==`c'
        lab val income_pct_w2_`c' income_pct`c'
    }

  * one variable including all countries
    gen income_pct_w2 = -7 if country==. | country==30
    foreach c in 11 12 13 14 15 16 17 18 19 20 23 25 28 29 {
        replace income_pct_w2 =  1  if  income_pct_w2_`c'== 1
```

```
        replace income_pct_w2 =  2  if  income_pct_w2_`c'== 2
        replace income_pct_w2 =  3  if  income_pct_w2_`c'== 3
        replace income_pct_w2 =  4  if  income_pct_w2_`c'== 4
        replace income_pct_w2 =  5  if  income_pct_w2_`c'== 5
        replace income_pct_w2 =  6  if  income_pct_w2_`c'== 6
        replace income_pct_w2 =  7  if  income_pct_w2_`c'== 7
        replace income_pct_w2 =  8  if  income_pct_w2_`c'== 8
        replace income_pct_w2 =  9  if  income_pct_w2_`c'== 9
        replace income_pct_w2 = 10  if  income_pct_w2_`c'==10
    }

    lab var income_pct_w2 "hhd income percentiles wave 2"
    lab val income_pct_w2 income_pct_w2
    lab def income_pct_w2 -7 "not yet coded", add

    keep mergeid fahcv_mod fohcv_mod income_pct_w2

save $easy\data\temp\sharew2_imputations_a.dta, replace
*>> w4:

use $easy\data\release\sharew4_cv_r.dta, clear
    // as year of interview is not included in w4 imputations
    merge m:m mergeid using $easy\data\release\sharew4_imputations.dta

    drop if mergeid =="no int w.4"
    keep if implicat==1

  * fahc: annual expenditure on food at home
    gen     fahcv_mod=fahc*excrate       if country !=35 // euro amounts into
                                                         // local currency
                                         // Estonia changed to Euro in 2011
    replace fahcv_mod=fahc*excrate       if country ==35 & int_year_w4==2010
    replace fahcv_mod=fahc               if country ==35 & int_year_w4==2011

    replace fahcv_mod=fahcv_mod/pppx2011 if  int_year_w4==2011
    replace fahcv_mod=fahcv_mod/pppx2010 if  int_year_w4==2010
    replace fahcv_mod=fahcv_mod/pppx2012 if  int_year_w4==2012
    replace fahcv_mod=fahcv_mod/12 // monthly instead of annual expenditure
    lab val fahcv_mod fahcv_mod
    lab var fahcv_mod "hhd monthly expenditure on food at home"

    *fohcv: expenditure on food outside home last 12 months
    gen     fohcv_mod=fohc*excrate       if country !=35 // euro amounts into
                                                         // local currency
                                         // Estonia changed to Euro in 2011
    replace fohcv_mod=fohc*excrate       if country ==35 & int_year_w4==2010
    replace fohcv_mod=fohc               if country ==35 & int_year_w4==2011

    replace fohcv_mod=fohcv_mod/pppx2010 if  int_year_w4==2010
    replace fohcv_mod=fohcv_mod/pppx2011 if  int_year_w4==2011
    replace fohcv_mod=fohcv_mod/pppx2012 if  int_year_w4==2012
    replace fohcv_mod=fohcv_mod/12 // monthly instead of annual expenditure
    lab val fohcv_mod fohcv_mod
    lab var fohcv_mod "hhd monthly expenditure on food outside home"


    *Income

    *thinc: household total net income (previous year)
    gen     thinc_m=thinc*excrate            // euro amounts into
                                             // local currency
            // even though Estonia chaned to Euro in 2011
            // the income measures going into thinc are mostly
            // based on the previous year, i.e. a non-Euro year
    replace thinc_m=thinc_m/pppx2010 if  int_year_w4==2010
    replace thinc_m=thinc_m/pppx2010 if  int_year_w4==2011
    replace thinc_m=thinc_m/pppx2011 if  int_year_w4==2012
```

```
    *country-specific income percentiles
    foreach c in 11 12 13 14 15 16 17 18 20 23 28 29 32 33 34 35 {
        pctile  p10_`c' = thinc_m if country==`c', nq(10)
        return list
        gen     income_pct_w4_`c'= 1 if thinc_m<=r(r1)                  ///
                                     & country==`c'
        replace income_pct_w4_`c'= 2 if thinc_m<=r(r2) & thinc_m>r(r1) ///
                                     & country==`c'
        replace income_pct_w4_`c'= 3 if thinc_m<=r(r3) & thinc_m>r(r2) ///
                                     & country==`c'
        replace income_pct_w4_`c'= 4 if thinc_m<=r(r4) & thinc_m>r(r3) ///
                                     & country==`c'
        replace income_pct_w4_`c'= 5 if thinc_m<=r(r5) & thinc_m>r(r4) ///
                                     & country==`c'
        replace income_pct_w4_`c'= 6 if thinc_m<=r(r6) & thinc_m>r(r5) ///
                                     & country==`c'
        replace income_pct_w4_`c'= 7 if thinc_m<=r(r7) & thinc_m>r(r6) ///
                                     & country==`c'
        replace income_pct_w4_`c'= 8 if thinc_m<=r(r8) & thinc_m>r(r7) ///
                                     & country==`c'
        replace income_pct_w4_`c'= 9 if thinc_m<=r(r9) & thinc_m>r(r8) ///
                                     & country==`c'
        replace income_pct_w4_`c'=10 if thinc_m>r(r9)                  ///
                                     & country==`c'
        lab val income_pct_w4_`c' income_pct`c'
    }

    *one variable including all countries
    gen income_pct_w4 =.
    foreach c in 11 12 13 14 15 16 17 18 20 23 28 29 32 33 34 35 {
        replace income_pct_w4 =  1  if  income_pct_w4_`c'== 1
        replace income_pct_w4 =  2  if  income_pct_w4_`c'== 2
        replace income_pct_w4 =  3  if  income_pct_w4_`c'== 3
        replace income_pct_w4 =  4  if  income_pct_w4_`c'== 4
        replace income_pct_w4 =  5  if  income_pct_w4_`c'== 5
        replace income_pct_w4 =  6  if  income_pct_w4_`c'== 6
        replace income_pct_w4 =  7  if  income_pct_w4_`c'== 7
        replace income_pct_w4 =  8  if  income_pct_w4_`c'== 8
        replace income_pct_w4 =  9  if  income_pct_w4_`c'== 9
        replace income_pct_w4 = 10  if  income_pct_w4_`c'==10
    }

    lab var income_pct_w4 "hhd income percentiles wave 4"
    lab val income_pct_w4 income_pct_w4
    label def income_pct_w4 -13 "not asked in this wave"
    format income_pct_w4 %38.0g

    keep mergeid fahcv_mod fohcv_mod income_pct_w4

save $easy\data\temp\sharew4_imputations_a.dta, replace


*------------------------------------------------------------------------------
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------

*----[23. Merge modules per wave ]---------------------------------------------

// We use the CV_R modules as master and then merge the other modules.

*>> w1:

use $easy\data\temp\sharew1_cv_r_b.dta     , clear

    assert mergeid!="no int w.1" // dropped non responding hh members above

    merge 1:1 mergeid using $easy\data\temp\sharew1_dn_a.dta,assert(  3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew1_ac_a.dta,assert(  3) nogen
```

```
    merge 1:1 mergeid using $easy\data\temp\sharew1_br_a.dta,assert(  3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew1_cf_a.dta,assert(  3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew1_ch_a.dta,assert(  3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew1_co_a.dta,assert(  3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew1_ep_a.dta,assert(  3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew1_hc_a.dta,assert(  3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew1_ph_a.dta,assert(  3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew1_sp_a.dta,assert(  3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew1_gv_health_a.dta        ///
                                                          ,assert(  3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew1_gv_isced_a.dta         ///
                                                          ,assert(  3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew1_dropoff_a.dta          ///
                                    ,  gen(merge_w1_dropoff) assert(1 3)
                                  // not every respondent has linked dropooff
    merge 1:1 mergeid using $easy\data\temp\sharew1_imputations_a.dta      ///
                                                          ,assert(1 3) nogen
      // specific imputated variables we want for easySHARE are not available
      // for IL (i.e. not included in "imputations_ilextra" module

save $easy\data\temp\sharew1_merged_a.dta, replace
*>> w2:

use $easy\data\temp\sharew2_cv_r_b.dta, clear

    assert mergeid!="no int w.2" // dropped non responding hh members above

    merge 1:1 mergeid using $easy\data\temp\sharew2_dn_a.dta,assert(  3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew2_ac_a.dta,assert(  3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew2_br_a.dta,assert(  3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew2_cf_a.dta,assert(  3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew2_ch_a.dta,assert(  3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew2_co_a.dta,assert(  3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew2_ep_a.dta,assert(  3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew2_hc_a.dta,assert(  3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew2_ph_a.dta,assert(  3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew2_sp_a.dta,assert(  3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew2_gv_health_a.dta        ///
                                                          ,assert(  3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew2_gv_isced_a.dta         ///
                                                          ,assert(  3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew2_imputations_a.dta      ///
                                                          ,assert(1 3) nogen
                // imputations for IE not available in SHARE main release
                // for 1 person in FR no imputations available

save $easy\data\temp\sharew2_merged_a.dta, replace

*>> w3:

use $easy\data\temp\sharew3_cv_r_b.dta, clear

    assert mergeid!="no int w.3" // dropped non responding hh members above

    merge 1:1 mergeid using $easy\data\temp\sharew3_st_a.dta,          nogen
                             // no assert because 66 cases in st not in cv_r
    merge 1:1 mergeid using $easy\data\temp\sharew3_cs_a.dta,assert(3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew3_hs_a.dta,assert(3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew3_gs_a.dta,assert(3) nogen

  * Adjustment in wave 3 for the missing 66 lines in w3 rel 1.0.0 cv_r file
    replace wave=3 if wave==.

save $easy\data\temp\sharew3_merged_a.dta, replace
```

```
*>> w4:

use $easy\data\temp\sharew4_cv_r_b.dta, clear

    assert mergeid!="no int w.4" // dropped non responding hh members above

    merge 1:1 mergeid using $easy\data\temp\sharew4_dn_a.dta,assert(3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew4_ac_a.dta,assert(3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew4_br_a.dta,assert(3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew4_cf_a.dta,assert(3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew4_ch_a.dta,assert(3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew4_co_a.dta,assert(3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew4_ep_a.dta,assert(3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew4_hc_a.dta,assert(3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew4_ph_a.dta,assert(3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew4_sn_a.dta,assert(3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew4_sp_a.dta,assert(3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew4_gv_health_a.dta   ///
                                                          ,assert(3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew4_gv_isced_a.dta    ///
                                                          ,assert(3) nogen
    merge 1:1 mergeid using $easy\data\temp\sharew4_imputations_a.dta ///
                                                          ,assert(3) nogen

save $easy\data\temp\sharew4_merged_a.dta, replace

*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------

*----[24. Other recodes per wave ]--------------------------------------------

*>> w1:

use $easy\data\temp\sharew1_merged_a.dta, clear

*>> Assign children information to partner of family respondent
    gsort coupleid1 -dumfamr
    replace ch001_  = ch001_[_n-1]       ///
                                if coupleid1==coupleid1[_n-1] & coupleid1!=""
    replace ch007_hh= ch007_hh[_n-1]     ///
                                if coupleid1==coupleid1[_n-1] & coupleid1!=""
    replace ch007_km= ch007_km[_n-1]     ///
                                if coupleid1==coupleid1[_n-1] & coupleid1!=""
    replace ch021_mod  = ch021_mod[_n-1] ///
                                if coupleid1==coupleid1[_n-1] & coupleid1!=""

*>> Assign variables on social support (received and given) to partner
    foreach var in sp002_ sp003_1 sp003_2 sp003_3 {
        replace `var' = `var'[_n-1] if coupleid1==coupleid1[_n-1] ///
                                    & coupleid1!="" & dumfamr==0 ///
                                    & dumfamr[_n-1]==1 & `var'==.
        replace `var' = `var'[_n-1] if coupleid1==coupleid1[_n-1] ///
                                    & coupleid1!="" & dumfamr==0 ///
                                    & `var'==. & `var'[_n-1]!=.
        replace `var' = 10 if `var' >= 10 & `var' < 20
    }

    foreach var in sp003_1 sp003_2 sp003_3 {
        replace `var' = -9 if sp002_== 5
    }

    foreach var in sp009_1 sp009_2 sp009_3 {
        replace `var' = -9  if sp008_ == 5
        replace `var' = 10 if `var' >= 10 & `var' < 20
    }
    lab def relative 10 "child" -9 "filtered: no help received/given", modify
```

```
    foreach var in sp003_1 sp003_2 sp003_3 sp009_1 sp009_2 sp009_3 {
        rename `var' `var'_mod
    }

    foreach var in sp002_ sp008_ {
        rename `var' `var'mod
    }

*>> Household able to make ends meet
    gen     co007a     =      co007_
    replace co007a     = . if co007a < 0
    egen    co007_sd =   sd(co007_), by(hhid1)
    replace co007a     = . if co007_sd > 0  &  co007_sd < .
    egen    co007b     =  min(co007a),by(hhid1)
    replace co007_     = co007b if co007b != .
    drop co007a co007b co007_sd

save $easy\data\temp\sharew1_merged_b.dta, replace

*>> w2:

use $easy\data\temp\sharew2_merged_a.dta, clear

*>> Assign children information to partner of family respondent
    gsort coupleid2 -dumfamr
    replace ch001_  =ch001_[_n-1]       ///
                                if coupleid2==coupleid2[_n-1] & coupleid2!=""
    replace ch007_hh=ch007_hh[_n-1]     ///
                                if coupleid2==coupleid2[_n-1] & coupleid2!=""
    replace ch007_km=ch007_km[_n-1]     ///
                                if coupleid2==coupleid2[_n-1] & coupleid2!=""
    replace ch021_mod=ch021_mod[_n-1]   ///
                                if coupleid2==coupleid2[_n-1] & coupleid2!=""

*>> Assign variables on social support (received and given) to partner
    foreach var in sp002_ sp003_1 sp003_2 sp003_3 {
        replace `var' = `var'[_n-1] if coupleid2==coupleid2[_n-1] ///
                                    & coupleid2!="" & dumfamr==0 ///
                                    & dumfamr[_n-1]==1 & `var'==.
        replace `var' = `var'[_n-1] if coupleid2==coupleid2[_n-1] ///
                                    & coupleid2!="" &dumfamr==0  ///
                                    & `var'==. & `var'[_n-1]!=.
        replace `var' = 10 if `var' >= 10 & `var' < 20
    }

    foreach var in sp003_1 sp003_2 sp003_3 {
        replace `var' = -9 if sp002_== 5
    }

    foreach var in sp009_1 sp009_2 sp009_3 {
        replace `var' = -9 if sp008_ == 5
        replace `var' = 10 if `var' >= 10 & `var' < 20
    }
    lab def relative 10 "child" -9 "filtered: no help received/given", modify

    foreach var in sp003_1 sp003_2 sp003_3 sp009_1 sp009_2 sp009_3 {
        rename `var' `var'_mod
    }

    foreach var in sp002_ sp008_ {
        rename `var' `var'mod
    }

*>> Household able to make ends meet
    gen     co007a     = co007_
    replace co007a     = . if co007a < 0
    egen    co007_sd = sd(co007_), by(hhid2)
    replace co007a     = . if co007_sd > 0  &  co007_sd < .
```

```
    egen    co007b     = min(co007a),by(hhid2)
    replace co007_     = co007b if co007b != .
    drop co007a co007b co007_sd


save $easy\data\temp\sharew2_merged_b.dta, replace

*>> w3:

use $easy\data\temp\sharew3_merged_a.dta, clear

*>> Fix gender missing data problems in w3/SHARELIFE
    // The problem comes form 66 missing lines in the cv_r dataset
    replace female = 0 if female==. & sl_st011_ == 1
    replace female = 1 if female==. & sl_st011_ == 2

save $easy\data\temp\sharew3_merged_b.dta, replace

*>> w4:

use $easy\data\temp\sharew4_merged_a.dta, clear

*>> Assign children, age and education to partner of the family respondent
    gsort   coupleid4 -dumfamr
    replace ch001_  =ch001_[_n-1]     ///
                            if coupleid4==coupleid4[_n-1] & coupleid4!=""
    replace ch007_hh=ch007_hh[_n-1]   ///
                            if coupleid4==coupleid4[_n-1] & coupleid4!=""
    replace ch007_km=ch007_km[_n-1]   ///
                            if coupleid4==coupleid4[_n-1] & coupleid4!=""
    replace ch021_mod=ch021_mod[_n-1] ///
                            if coupleid4==coupleid4[_n-1] & coupleid4!=""

*>> Assign variables on social support (received and given) to partner
    foreach var in sp002_ sp008_ {
        replace `var' = `var'[_n-1] if coupleid4==coupleid4[_n-1] ///
                                    & coupleid4!="" & dumfamr==0 ///
                                    & dumfamr[_n-1]==1 & `var'==.
        replace `var' = `var'[_n-1] if coupleid4==coupleid4[_n-1] ///
                                    & coupleid4!="" & dumfamr==0 ///
                                    & `var'==. & `var'[_n-1]!=.
    }

    foreach var in sp003_1 sp003_2 sp003_3 sp009_1 sp009_2 sp009_3 {
        rename `var' `var'_mod
    }

    foreach var in sp002_ sp008_ {
        rename `var' `var'mod
    }

*>> Household able to make ends meet
    gen     co007a     = co007_
    replace co007a     = . if co007a < 0
    egen    co007_sd = sd(co007_), by(hhid4)
    replace co007a     = . if co007_sd > 0  &  co007_sd < .
    egen    co007b     = min(co007a),by(hhid4)
    replace co007_     = co007b if co007b != .
    drop    co007a co007b co007_sd


save $easy\data\temp\sharew4_merged_b.dta, replace
```

```
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------

*----[25. Append waves to panel long format & integrate "long" variables]------

*>> Append single wave files to one long file:

    use          $easy\data\temp\sharew1_merged_b.dta, clear
    append using $easy\data\temp\sharew2_merged_b.dta
    append using $easy\data\temp\sharew3_merged_b.dta
    append using $easy\data\temp\sharew4_merged_b.dta


*>> Integrate hhid? into a "long" format hidd variable
    gen      hhid = ""
    foreach w in $w {
        replace hhid = hhid`w' if wave==`w'
    }
    drop     hhid?
    lab var  hhid "household identifier in respective wave - see var. wave"


*>> Integrate coupleID? into a "long" format coupleid variable
    gen      coupleid = ""
    foreach w in $w {
     replace coupleid = coupleid`w' if wave==`w'
    }
    drop     coupleid?
    lab var  coupleid "couple identifier in respective wave - see var. wave"


*>> Generate wave participation overview variable
    foreach w in $w {
        gen temp1_`w' = `w' if wave==`w'
       egen temp2_`w' = max(temp1_`w'), by(mergeid)
    }
    gen      wavepart = ""
    foreach w in $w {
     replace wavepart = wavepart + string(temp2_`w')  if string(temp2_`w')!="."
    }
    destring wavepart, replace
    lab var  wavepart "wave participation pattern"

    drop temp1_* temp2_*

*------------------------------------------------------------------------------
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------

*----[26. Fix date intv., year/month birth, gender, & partnervars & gen age]---

*   Using information of other waves and simple mode imputations for date of
*   interview and month of birth (if year is missing in all waves available,
*   we do not impute)

*>> Impute date of interview if missing (SHARELIFE known issue)
*   by (minimum) mode values of wave/country:
    gen     int_date = ym(int_year, int_month) // %tm format
    replace int_date =dofm(int_date)           // changes to date format

    egen   int_date_mode = mode(int_date),by(wave country_mod) minmode
    format int_date_mode int_date %d

    replace int_month = month(int_date_mode)  if int_month==.
    replace int_year  =  year(int_date_mode)  if int_year    ==.

    drop int_date int_date_mode
```

```
*>> Check for deviations within gender (known issue in IL wave 1 vs 2):
    // if gender deviates between waves, one information must be wrong
    // as there is no way to know which is the wrong information, both
    // are set to -3 "implausible value/suspected wrong"
    egen    female_sd = sd(female), by(mergeid)
    replace female = -3 if female_sd > 0 & female_sd < .
    drop                   female_sd

*>> Check for deviations within year of birth (known issue in IL wave 1 vs 2):
    // same as with gender:
    gen     dn003_mod  = dn003_ if dn003_ > 0

    egen    dn003_mod_sd = sd(dn003_mod), by(mergeid)
    replace dn003_mod    = -3 if dn003_mod_sd > 0 & dn003_mod_sd < .
    replace dn003_mod    = -3 if dn003_mod == 2011
    drop                   dn003_mod_sd

    lab var dn003_mod "year of birth"
    lab def dn003_mod -15 "no information"                    ///
                       -3 "implausible value/suspected wrong"
    lab val dn003_mod dn003_mod

*>> Generate new month of birth variable, taking the minimum mode of all info:
    // here we are less strict as with year of birth and gender
    // i.e. we do not set the self-report to missing if it deviates between
    // waves, instead we take the minimum modus answer of the self-reported
    // dn002_ variable per person:

    gen     dn002_mod = dn002_ if dn002_ > 0
    egen    dn002_mod_modus = mode(dn002_mod), by(mergeid) minmode
    replace dn002_mod = dn002_mod_modus if dn002_mod != dn002_mod_modus
    drop                dn002_mod_modus

    lab val dn002_mod month
    lab var dn002_mod "month of birth"


*>> Now we can replace missing information from another wave if available:
    foreach var in female dn003_mod dn002_mod {
      di "`var'"
        // Here we start to copy the information to all lines of
        // the respondent; i.e. isced_r from wave 1 is written into
        // all lines of the same respondent.
        sort mergeid
        foreach i in $w {
           di "`var'"
            replace `var' = `var'[_n+`i'] if mergeid==mergeid[_n+`i'] & ///
                                             `var'== . & `var'[_n+`i'] !=.
            replace `var' = `var'[_n-`i'] if mergeid==mergeid[_n-`i'] & ///
                                             `var'== . & `var'[_n-`i'] !=.
        }
    }


*>> Finally we can generate age of respondent:
    * for the ~200 respondents with available year of birth (dn003) but missing
    * month of birth (dn002), we assume they are born in June:

    gen     age = ( (int_year * 12  + int_month) -                    ///
                    (dn003_mod * 12 + dn002_mod) ) / 12 if dn003_mod != -3

    replace age = ( (int_year * 12  + int_month) -                    ///
                      (dn003_mod * 12 + 6        ) ) / 12 if dn003_mod != -3 ///
                                                     & dn002_mod ==  .

    lab var  age "age at interview (in years)"
```

```
*>> Generate Age of partner

    gen age_partner = .
    // if we have a self report of the repective wave specific partner
    // we take it:
    sort coupleid
    replace age_partner = age[_n-1] if coupleid==coupleid[_n-1] & coupleid!=""
    replace age_partner = age[_n+1] if coupleid==coupleid[_n+1] & coupleid!=""

    // otherwise we generate age based on the cv_r information:
    replace age_partner = ( (int_year * 12        + int_month      ) -       ///
                            (yrbirth_partner * 12 + mobirth_partner) ) / 12  ///
                           if age_partner == .        /// no self-report
                            & coupleid != ""          /// has partner
                            & yrbirth_partner > 0 & yrbirth_partner < .      ///
                            & mobirth_partner > 0 & mobirth_partner < .

    // finally, as with age we assume the partner is born in June if only
    // the month of birth of partner is missing, but the year is available:
    replace age_partner = ( (int_year * 12        + int_month      ) -       ///
                            (yrbirth_partner * 12 + 6              ) ) / 12  ///
                            if age_partner == .        /// no self-report
                             & coupleid != ""          /// has partner
                             & yrbirth_partner > 0 & yrbirth_partner < .

    drop yrbirth_partner mobirth_partner

    assert age_partner == . if coupleid=="" // just to make sure
    assert age_partner >= 0

    replace age_partner =  -9 if    coupleid==""
    replace age_partner = -15 if age_partner== .

    lab var age_partner "age at interview of partner"

    lab def age_partner -15 "no information"                          ///
                         -9 "filtered: single or no partner in hh"    ///
                         -3 "implausible value/suspected wrong"
    lab val age_partner age_partner

*>> Take out very weird values in age & age_partner
    replace age         = -3 if age              < 14
    replace age_partner = -3 if age_partner < 14 & age_partner >=0

*>> After having transferred age to age_partner in couples set no info codes
*   in dn003_mod dn002_mod and age:

    replace dn003_mod = -15 if dn003_mod == .
    replace dn002_mod = -15 if dn002_mod == .

    replace age =   -3 if dn003_mod == -3
    replace age = - 15 if       age == .

    lab def age -15 "no information"                          ///
                -3 "implausible value/suspected wrong"
    lab val age age


*>> Change storage format of age and age_partner to only one decimal
    // it is only month exact

    replace age         = round(age        ,0.1)
    replace age_partner = round(age_partner,0.1)
    format  age         %9,1f
    format  age_partner %9,1f

*>> Correct gender_partner from self-reports otherwise take cv_r variable:
```

```
    sort coupleid
    replace gender_partner = female[_n-1] if coupleid==coupleid[_n-1] ///
                                          & coupleid!=""
    replace gender_partner = female[_n+1] if coupleid==coupleid[_n+1] ///
                                          & coupleid!=""

    replace gender_partner =  -9 if          coupleid==""
    replace gender_partner = -15 if gender_partner== .

    lab var gender_partner "gender of partner: female=1, male=0"
    lab def gender_partner -15 "no information"                              ///
                            -9 "filtered: single or no partner in hh"        ///
                            -3 "implausible value/suspected wrong"           ///
                             0 "male"                                        ///
                             1 "female"
    lab val gender_partner gender_partner

*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------

*----[27. Transfer information collected once (in baseline interviews)]--------
*         to the next waves of the respondent
*         (done for isced, eduyears, country of birth, citizenship, height)

foreach var in isced_r eduyears birth_country citizenship ph013_ {
  di "`var'"
    // First we check for deviations within person and set the
    // variable to missing if we have contradictory information.
    // This occurs rarely and there is no way to know which
    // information is correct.
    gen     `var'c   = `var'
    replace `var'c   = .  if `var'c < 0
    egen    `var'c_sd =   sd(`var'c), by(mergeid)
    replace `var'c    = . if `var'c_sd > 0  &  `var'c_sd < .

    // Here we start to copy the information to all lines of
    // the respondent; i.e. isced_r from wave 1 is written into
    // all lines of the same respondent.
    sort mergeid
    foreach i in $w {
       di "`var'"
        replace `var'c = `var'c[_n+`i'] if mergeid==mergeid[_n+`i'] & ///
                        `var'c== . & `var'c[_n+`i'] !=.
        replace `var'c = `var'c[_n-`i'] if mergeid==mergeid[_n-`i'] & ///
                        `var'c== . & `var'c[_n-`i'] !=.
    }
    // Here we copy back the missing codes, only into the line of the wave
    // the missing code occurred, if we did not find a valid answer in any
    // other wave.
    replace `var'c = `var' if `var'c == . & `var' < 0
    rename      `var' `var'_orig // to preserve the label stored in `var'
    rename  `var'c `var'
}

lab val isced_r isced
lab var isced_r "education of respondent in ISCED-97 Coding"

lab val eduyears dn041
lab var eduyears "years of education"
rename  eduyears eduyears_mod // We ignored the coded eduyears from wave 1
                              // in this variable, this is why we add the
                              // "modified" indicator here.

drop isced_r_orig eduyears_orig isced_rc_sd eduyearsc_sd

foreach var in isced_r eduyears_mod {
    lab def `var'  95 "still in school", add
```

```
    lab def `var'  97 "other", add
}

lab val citizenship     citizenship     // labelname of dn008c
lab var citizenship     "citizenship of respondent (ISO coded)"

lab val birth_country   countryofbirth  // labelname of dn005c
lab var birth_country   "country of birth (ISO coded)"

lab def countryofbirth  1095 "Former Protectorate of Northern Rhodesia", add

drop dn008c dn005c // Now we don't need these two w4 variables anymore.
                   // They were only kept to have their label info available.

*------------------------------------------------------------------------------
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------

*----[28. Pass on information to next wave that may have changed/not changed]--


replace wave=33 if wave==3 // Change wave variable to 33 for SHARELIFE
                           // This allows sorting w1, w2 and w4 in order
                           // and to easier omitt wave 3.


*>> Assign marital status if longitudinal
    sort mergeid wave
    replace dn014_ = dn014_[_n-1] if mergeid==mergeid[_n-1] & dn044_ ==  5 ///
                                                            & wave   != 33
    rename  dn014_ mar_stat
    drop        dn044_


*>> Assign siblings and parents alive:
    sort mergeid wave
    replace siblings_alive = -9                     ///
                    if siblings_alive == .          ///
                      & mergeid == mergeid[_n-1]    ///
                      & (siblings_alive[_n-1] == 0  ///
                      | siblings_alive[_n-1] == -9) ///
                      & wave != 33
    replace siblings_alive = - 3 if siblings_alive==99

    replace dn026_1=5 if mergeid==mergeid[_n-1]                         ///
                       & (dn026_1==. | dn026_1<0) & dn026_1[_n-1]==5    ///
                       & wave!=33
    replace dn026_2=5 if mergeid==mergeid[_n-1]                         ///
                       & (dn026_2==. | dn026_1<0) & dn026_2[_n-1]==5    ///
                      & wave!=33

    rename  dn026_1 mother_alive
    rename  dn026_2 father_alive


*>> Assign ever smoked and currently smoking
    sort mergeid wave
    replace br001_=5 if mergeid==mergeid[_n-1] & br001_[_n-1]==5 ///
                      & br001_==. & wave!=33
    replace br001_=1 if mergeid==mergeid[_n-1] & br001_[_n-1]==1 ///
                      & br001_==. & wave!=33
    replace br002_=5 if br001_==5 & br002_==.
    rename  br001_ ever_smoked
    rename  br002_ smoking
    label   def br002 5 "no", modify

*>> Assign children living in same hh or w4
    sort mergeid wave
```

```
    replace ch007_hh=ch007_hh[_n-1] if mergeid==mergeid[_n-1] & ch524_==5 ///
                                    & wave !=33
    replace ch007_km=ch007_km[_n-1] if mergeid==mergeid[_n-1] & ch524_==5 ///
                                    & wave !=33


replace wave=3 if wave==33 // Change wave variable back to 3 for SHARELIFE

*------------------------------------------------------------------------------
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------

*----[29. Fix & re-generate variables, labels, etc.]---------------------------


*>> Use information from social networks module to reduce missing values
*   in mother-/father_alive in wave 4
    foreach var in sn005_1 sn005_2 sn005_3 sn005_4 sn005_5 sn005_6 sn005_7 {
        replace mother_alive = 1 if `var' ==2
    }

    foreach var in sn005_1 sn005_2 sn005_3 sn005_4 sn005_5 sn005_6 sn005_7 {
        replace father_alive = 1 if `var' ==3
    }

*>> Re-generate yes/no values (1=yes, 5=no) for consistency reasons
    replace ch007_hh = 5 if ch007_hh ==0
    lab def lblch007_hh 5 "no", modify

    replace ch007_km = 5 if ch007_km ==0
    lab def lblch007_km 5 "no", modify

    replace hc032c = 5 if hc032c ==0
    lab def lblhc032c 5 "no", modify

*>> Calculate BMI using the transferred height (ph013) and ph012
*   (collected in every wave)
    // correct obvious typos in height (Iwer typed in meters not centimetres):
    replace ph013_ = ph013_ * 100 if ph013_ >= 1.00 & ph013_ <= 1.90

    gen       bmi_mod = (ph012_/(ph013_*ph013_)) * 10000 if ph012_>-1 & ph013_>-1
    replace bmi_mod = -12 if ph012_==-1 | ph012_==-2
    replace bmi_mod = -12 if ph013_==-1 | ph013_==-2
    replace bmi_mod = -15 if ph012_==.   | ph013_==.
    recode  bmi_mod (0/12        = -3)
    recode  bmi_mod (100/.       = -3)

    recode  bmi_mod (  12/18.499999=1)   ///
                    (18.5/24.999999=2)  ///
                    (  25/29.999999=3)  ///
                    (  30/100=4)          , gen(bmi2_mod)
    replace bmi2_mod = -3 if bmi_mod == -3

    lab var bmi_mod    "body mass index (easySHARE version)"
    lab var bmi2_mod   "body mass index categories (easySHARE version)"

    lab def bmi_mod                          ///
          -3 "value suspected wrong"
    lab val bmi_mod bmi_mod

    lab def bmi2_mod                         ///
           1 "below 18.5 -underweight"       ///
           2 "18.5 - 24.9 - normal"          ///
           3 "25-29.9 -overweight"           ///
           4 "30 and above -obese"           ///
          -3 "value suspected wrong"
    lab val bmi2_mod bmi2_mod
```

```
*>> Add labels for missing imputations in IL wave 1:

    replace income_pct_w1 =-7 if wave==1 & country==25
    lab def income_pct_w1  -7 "not imputed yet", add

    replace fohcv_mod=-7 if wave==1 & country==25
    lab def fohcv_mod -7 "not imputed yet", add

    replace fahcv_mod=-7 if wave==1 & country==25
    lab def fahcv_mod -7 "not imputed yet", add

*>> Add labels for country
    lab def country 32 "Hungary",add
    lab def country 33 "Portugal",add
    lab def country 34 "Slovenia",add
    lab def country 35 "Estonia",add

*>> Add labels for language
    lab def language 32 "Hungarian",add
    lab def language 33 "Portuguese",add
    lab def language 34 "Slovenian",add
    lab def language 35 "Estonian or Russian (ee)",add

*>> Add, re-assign or complete missing labels
    lab var casp "CASP: quality of life and well-being index"
    lab val casp casp

    lab val partnerinhh partnerinhh
    lab def partnerinhh 1 "living with a spouse/partner in household"       ///
                        3 "living without a spouse/partner in household"    ///
                       97 "other"

    lab var partnerinhh "living with spouse/partner"

    lab val hhsize      hhsize
    lab val recall_1    recall_1
    lab val recall_2    recall_2
    lab val chronic_mod chronic_mod
    lab val mobilityind mobilityind
    lab val lgmuscle    lgmuscle
    lab val adlwa       adlwa
    lab val adla        adla
    lab val grossmotor  grossmotor
    lab val finemotor   finemotor
    lab val iadla       iadla
    lab val iadlza      iadlza
    lab val maxgrip     maxgrip

    lab def hhsize      -15 "no information"
    lab def recall_1    -15 "no information"
    lab def recall_2    -15 "no information"
    lab def chronic_mod -15 "no information"
    lab def mobilityind -15 "no information"
    lab def lgmuscle    -15 "no information"
    lab def adlwa       -15 "no information"
    lab def adla        -15 "no information"
    lab def grossmotor  -15 "no information"
    lab def finemotor   -15 "no information"
    lab def iadla       -15 "no information"
    lab def iadlza      -15 "no information"
```

```
*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------

*----[30. Implement/complete wave/country skip patterns]----------------------

    foreach var in mar_stat mother_alive father_alive siblings_alive {
        replace `var' = -13 if wave==3
    }

    foreach var in co007_ ever_smoked smoking br010_mod br015_ ///
                   recall_1 recall_2 {
        replace `var' = -13 if wave==3
    }

    foreach var in ch001_ ch021_mod bmi_mod bmi2_mod chronic_mod {
        replace `var' = -13 if wave==3
    }

    foreach var in ep005_ ep009_ ep011_mod ep013_mod ep026_mod ep036_mod {
        replace `var' = -13 if wave==3
    }

    foreach var in hc002_ hc012_ hc029_ {
        replace `var' = -13 if wave==3
    }

    foreach var in sp002_  sp003_1 sp003_2 sp003_3 sp009_1 sp009_2      ///
                   sp009_3 sp008_ {
        replace `var' = -13 if wave==3
    }

    foreach var in casp euro1 euro2 euro3 euro4 euro5 euro6 euro7 euro8 ///
                   euro9 euro10 euro11 euro12 eurod {
        replace `var' = -13 if wave==3
    }

    foreach var in mobilityind lgmuscle adlwa adla grossmotor finemotor ///
                   iadla iadlza orienti numeracy {
        replace `var' = -13 if wave==3
    }

    foreach var in ac002d1 ac002d2 ac002d3 ac002d4 ac002d5 ac002d6      ///
                   ac002d7 ac002dno {
        replace `var' = -13 if wave==3 | wave==4
    }

    foreach var in ch007_hh ch007_km {
        replace `var' = - 7 if wave==4 & (ch007_hh==. | ch007_km==.)
        replace `var' = -13 if wave==3
    }

    foreach var in hc038_mod hc032c {
        replace `var' = -13 if wave==3 | wave==4
    }

    foreach var in   q34_re {
        replace `var' = -13 if wave==2 | wave==3 | wave==4
    }


    foreach var in fahcv_mod fohcv_mod {
        replace `var' = -13 if wave==3
        replace `var' = -7 if country == 30
    }

    replace income_pct_w1 = -13 if wave !=1
    replace income_pct_w2 = -13 if wave !=2
```

```
    replace income_pct_w2 = -7   if country ==30
    replace income_pct_w4 = -13 if wave !=4


*>> Variables only included in SHARELIFE
    foreach var in sl_cs008_ sl_cs009_ sl_cs010_mod sl_cs010a_mod {
        replace `var' = -13 if wave==1 | wave==2 | wave==4
    }

*>> q34_re : religion not asked in France drop-off wave 1:

    replace q34_re =-14 if substr(mergeid,1,2)=="FR" & wave==1


*>> Set additional missing code if information is not available
*   because wave 1 dropoff is not available / could not be linked

    foreach var in cesd1 casp q34_re {
        replace `var' = -16 if merge_w1_dropoff == 1 & wave==1
    }

    lab def cesd1 -16                                              ///
        "no drop-off (information in drop-off in this wave)"

    lab val cesd1 cesd1

    foreach label in casp reli cesd1 {
      lab def `label' -16                                          ///
        "no drop-off (information in drop-off in this wave)"       ///
      , add modify
    }

    drop merge_w1_dropoff

*------------------------------------------------------------------------------
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------

*----[31. Integrate DK/RF and implement no information missing code]-----------


*>> Integrate "don't know" and "refusal" for remaining variables
    foreach var in                                                      ///
        female age partnerinhh dn002_ dn003_ gender_partner age_partner     ///
        hhsize country country_mod birth_country citizenship sphus          ///
        sl_cs008_ sl_cs009_ sl_cs010_mod sl_cs010a_mod                       ///
        q34_re                                                              ///
        mar_stat mother_alive father_alive siblings_alive                   ///
        fahcv_mod fohcv_mod income_pct_w1 income_pct_w2 income_pct_w4       ///
        mobilityind lgmuscle adlwa adla grossmotor finemotor                ///
        iadla iadlza orienti numeracy maxgrip eurod                         ///
        isced_r eduyears_mod                                                ///
        co007_ ever_smoked smoking br010_mod br015_ recall_1 recall_2       ///
        ac002d1 ac002d2 ac002d3 ac002d4 ac002d5 ac002d6 ac002d7             ///
        ac002dno                                                            ///
        sp002_ sp003_1 sp003_2 sp003_3 sp008_ sp009_1 sp009_2 sp009_3       ///
        ch001_ ch021_mod bmi_mod chronic_mod                                ///
        ch007_hh ch007_km                                                   ///
        ep005_ ep009_ ep011_mod ep013_mod ep026_mod ep036_mod               ///
        hc002_ hc012_ hc029_ hc038_mod hc032c                               ///
        casp euro1 euro2 euro3 euro4 euro5 euro6 euro7 euro8               ///
        euro9 euro10 euro11 euro12 eurod                                {

            replace `var' = -12 if `var'==-1 | `var'==-2

    }
```

```
*>> Set "no information" missing code, when no other reason is left:
    foreach var in                                                          ///
            female age dn002_ dn003_ partnerinhh                            ///
            gender_partner age_partner int_partner                          ///
            hhsize country country_mod birth_country citizenship sphus      ///
            q34_re                                                          ///
            sl_cs008_ sl_cs009_ sl_cs010_mod sl_cs010a_mod                   ///
            mar_stat mother_alive father_alive siblings                     ///
            fahcv_mod fohcv_mod income_pct_w1 income_pct_w2 income_pct_w4    ///
            mobilityind lgmuscle adlwa adla grossmotor finemotor            ///
            iadla iadlza orienti numeracy maxgrip eurod                     ///
            isced_r eduyears_mod                                            ///
            co007_ ever_smoked smoking br010_mod br015_ recall_1 recall_2   ///
            ac002d1 ac002d2 ac002d3 ac002d4 ac002d5 ac002d6 ac002d7         ///
            ac002dno                                                        ///
            sp002_ sp003_1 sp003_2 sp003_3 sp008_ sp009_1 sp009_2 sp009_3   ///
            ch001_ ch021_mod bmi_mod chronic_mod                            ///
            ch007_hh ch007_km                                               ///
            ep005_ ep009_ ep011_mod ep013_mod ep026_mod ep036_mod           ///
            hc002_ hc012_ hc029_ hc038_mod hc032c                           ///
            casp euro1 euro2 euro3 euro4 euro5 euro6 euro7 euro8            ///
            euro9 euro10 euro11 euro12 eurod                          {

                replace `var' = -15 if `var'==.
    }

*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------
*-------------------------------------------------------------------------------

*----[32. Keep, add easy missing codes & labels, order, data labels & save]----

keep  mergeid hhid coupleid wave wavepart                                   ///
      int_year int_month country country_mod language                       ///
      female dn002_mod dn003_mod age birth_country citizenship q34_re       ///
      isced_r eduyears_mod mar_stat hhsize partnerinhh int_partner          ///
      age_partner gender_partner  mother_alive father_alive siblings_alive  ///
      ch001_ ch021_mod ch007_hh ch007_km                                    ///
      ac002d1 ac002d2 ac002d3 ac002d4 ac002d5 ac002d6 ac002d7 ac002dno      ///
      sp002_ sp003_1 sp003_2 sp003_3 sp008_ sp009_1 sp009_2 sp009_3         ///
      sl_cs008_ sl_cs009_ sl_cs010_mod sl_cs010a_mod                        ///
      sphus chronic_mod casp euro1 euro2 euro3 euro4 euro5 euro6 euro7      ///
      euro8 euro9 euro10 euro11 euro12 eurod                                ///
      hc002_ hc012_ hc029_ hc038_mod hc032c                                 ///
      maxgrip adlwa adla iadla iadlza mobilityind lgmuscle                  ///
      grossmotor finemotor recall* orienti numeracy                         ///
      bmi_mod bmi2_mod smoking ever_smoked br010_mod br015_                 ///
      ep005_ ep009_ ep011_mod ep013_mod ep026_mod ep036_mod co007_          ///
      fahcv_mod fohcv_mod income_pct_w1 income_pct_w2 income_pct_w4

      // cesd1 not included in this release - needs more checks

*>> Add additional easy labels to all remaining labelsets:
    label dir
    global labellist "`r(names)'"
    foreach label in $labellist {
        lab def `label'  -3 "implausible value/suspected wrong", add modify
        lab def `label'  -7 "not yet coded"                     , add modify
        lab def `label' -12 "don´t know / refusal"              , add modify
        lab def `label' -13 "not asked in this wave"            , add modify
        lab def `label' -14 "not asked in this country"         , add modify
        lab def `label' -15 "no information"                    , add modify
    }
    lab def maxgrip -15 "no/no valid measure"
    lab val maxgrip maxgrip

*>> Order
order mergeid hhid coupleid wave wavepart                                   ///
```

```
      int_year int_month country country_mod language                          ///
      female dn002_mod dn003_mod age birth_country citizenship q34_re          ///
      isced_r eduyears_mod mar_stat hhsize partnerinhh int_partner             ///
      age_partner gender_partner  mother_alive father_alive siblings_alive     ///
      ch001_ ch021_mod ch007_hh ch007_km                                       ///
      ac002d1 ac002d2 ac002d3 ac002d4 ac002d5 ac002d6 ac002d7 ac002dno         ///
      sp002_ sp003_1 sp003_2 sp003_3 sp008_ sp009_1 sp009_2 sp009_3            ///
      sl_cs008_ sl_cs009_ sl_cs010_mod sl_cs010a_mod                           ///
      sphus chronic_mod casp euro1 euro2 euro3 euro4 euro5 euro6 euro7         ///
      euro8 euro9 euro10 euro11 euro12 eurod                                   ///
      hc002_ hc012_ hc029_ hc038_mod hc032c                                    ///
      maxgrip adlwa adla iadla iadlza mobilityind lgmuscle                     ///
      grossmotor finemotor recall* orienti numeracy                            ///
      bmi_mod bmi2_mod smoking ever_smoked br010_mod br015_                    ///
      ep005_ ep009_ ep011_mod ep013_mod ep026_mod ep036_mod co007_             ///
      fahcv_mod fohcv_mod income_pct_w1 income_pct_w2 income_pct_w4

    sort mergeid wave

*>> Label & Notes for dataset
    label data ""
    notes drop _dta
    note: {it: easy}SHARE release {cmd: 1.0.0} waves {cmd: $w} ///
          {it:doi } {cmd: 10.6103/SHARE.easy.100}
    label data "easySHARE release 1.0.0  waves $w  doi 10.6103/SHARE.easy.100 "

compress

saveold $easy\data\easySHARE_rel1-0-0.dta  // no replace on purpose!
                                          // this is to make sure
                                          // we do not delete a
                                          // released version by
                                          // accident
    // saveold is used to have the dataset readable in Stata 11


*>> Use this file to convert it to SPSS
*   so that we do not have double numlabels (SPSS has an option to show them):

   *>> first write an SPSS script to run from within SPSS:
    clear
    set obs 5
    gen str60 var = ""
    replace var= "GET STATA FILE='$easy/data/easySHARE_rel1-0-0.dta'. " in 1
    replace var= "SAVE OUTFILE  ='$easy/data/easySHARE_rel1-0-0.sav'  " in 2
    replace var= " /COMPRESSED.                                       " in 3

    outfile using "$easy/data/Save_as_spss.sps", replace  noquote

   *>> execute SPSS
    // $spss_executable is a global that must be set for each computer)
    //   and points to the SPSS executable
    // Alternatively one can just start SPSS from the Start menue and
    //   load the Save_as_spss.sps script just created above.

    quietly {

    . winexec $spss_executable "$easy/data/Save_as_spss.sps"
    window stopbox note ///
        "Run file in SPSS (press STRG+A to mark all, then STRG+R to run)." " " ///
        "Close SPSS completely and then press OK to continue."
    }

   *>> delete the SPSS script:
    erase $easy/data/Save_as_spss.sps
```

```
*>> Only now we add the numerical codes into the label for the stata users

use $easy\data\easySHARE_rel1-0-0.dta, clear

    numlabel, add force

*>> And replace the version we just stored before:
saveold $easy\data\easySHARE_rel1-0-0.dta, replace

*------------------------------------------------------------------------------
*------------------------------------------------------------------------------
*------------------------------------------------------------------------------

*----[  . Zip-archive the release files used]----------------------------------
*>> Archive the release files used

cd $easy/data/release
zipfile *.dta,
saving("$easy/data/easySHARE_${release}_ReleaseFilesUsed_`c(current_date)'.zip")


*>> Archive the temp data files used

cd $easy/data/temp
zipfile *.dta, ///
 saving("$easy/data/easySHARE_${release}_TempFilesUsed_`c(current_date)'.zip")

*>> Archive easyDatasets

cd $easy/data/
zipfile "$easy/data/easySHARE_rel1-0-0.dta" ///
        "$easy/data/easySHARE_rel1-0-0.sav", ///
 saving("$easy/data/easySHARE_${release}_FinalDatasets_`c(current_date)'.zip")

*>> Now the following directories can be deleted (manually):

    * $easy/data/temp
    * $easy/data/release

    quietly {

    window stopbox note ///
        "The following directories have been zipped and can be deleted." ///
        "${easy}/data/temp"                                              ///
        "${easy}/data/release"
    }


*----[  . Exit Stata & Close Log File]----------------------------------------

capture log close
exit
```