PISA INTERNATIONAL DATA BASE

July 31, 2001

Project Consortium:

Australian Council For Educational Research (ACER)

Netherlands National Institute for Educational Measurement (CITO group)

Educational Testing Service (ETS)

National Institute for Educational Policy Research (NIER, Japan)

Westat

TABLE OF CONTENTS

BACKGROUND

1. At the beginning of May 2001, the first release of the international database was returned to the PISA 2000 National Centres. When distributed, it was clearly stated that this current version had been prepared for review purposes and to permit national centres to undertake preliminary analyses.

2. The next sections present the structure of the international database as released at the end of July 2001.

3. The differences between the May release and the July release are listed hereafter.

- The data for Latvia are now included in the current version.
- The current international database consists of seven files, instead of 5 files in the previous version. The student background information and the proficiency variables are now provided in three different files, labelled "instui_read_0701.txt", "intstui_math_0701.txt" and "intstui_scie_0701.txt". It was decided to provide three different files to avoid the recoding of the student final weight and Fay's replicate for analysing the mathematics and science data. Therefore, the mathematics adjustment factor and the science adjustment factor are not included any more, as the weights and Fay's replicate have already been adjusted.
- To avoid misuses of the data, the mathematics and science achievement variables are not included in the reading file "intstui_read_0701.txt". The science achievement variables are not included in the mathematics file "intstui_math_0701.txt" and the mathematics achievement variables are not included in the science file "intstui_scie_0701.txt". If one want to compute the correlation between mathematics and reading, the mathematic file should be used. Also, the correlation between science and reading can be computed by using the science file. It is worth noting that the mathematics file consists of only students who were assessed in this particular domain. Also, the science file consists of the subset of students who were assessed in science.
- The student files contain the Cross-Curriculum Competencies (CCC) and Information Technologies familiarity (IT) composites. A description of these composites is provided later.
- Cross-countries validity analyses were performed on the student and school composites, included the CCC and IT composites. These analyses have shown that some of the school and student composites presented a lack of consistencies across countries. The consortium, with the support of the TAG, has decided to not include any more these composites. These composites are OTHERSUP, REMSCH, EXTOUT, REMOUT, SCREUSE, USEHOME, DISENG, ABSENCE, SELECT, TRANSFER, SCREMRES, ASSCOMP and ASSDECI. Most of these composites are related to assessment practices, selection or transfer policies, and remedial or extension courses. The lack of variability within countries regarding these issues is probably one of the major reasons why these composites did not work.
- Three composites were changed. For ACHPRESS, item St26Q01 was dropped from the composite, for HMWKTIME, item st32Q01 was omitted and SCHAUTON has been inverted.
- A country adjustment factor is included within the three student files. These adjustment factors should be used to standardise the weights so that each country contributes equally to the international analyses on the marginal distributions. These adjustment factors should be applied on the student weights and on the Fay's replicates.
- The measurement error for the cognitive weighted maximum likelihood estimates are now included in the database.
- The stratification and school identification variables have been concatenated to form a unique school identification variable, labelled SCHOOLID. Nevertheless, one can still slip the new variable into the two previous one by modifying the SAS or SPSS syntax files.
- A few recodings were implemented on some national data sets on the request of some NPMs.

GENERAL STRUCTURE OF THE INTERNATIONAL DATA BASE

FILES INCLUDED IN THE DATABASE

4. The PISA international database consists of seven data files. The files are returned in text (or ASCII format) with the corresponding SAS and SPSS control files.

- Intstui_read_0701.txt consists of :
  - the student context variables collected through the student questionnaires,
  - the student composites derived from the original student context variables and the two international options (i.e. CCC and IT instrument),
  - the student achievement scores in reading,
  - the student reading weights and a country adjustment factor for the reading weights and Fay's replicate,
  - The reading Fay's replicate for the computation of the sampling variance estimates.

- Intstui_math_0701.txt consists of :
  - the student context variables collected through the student questionnaires,
  - the student composites derived from the original student context variables and the two international options (i.e. CCC and IT instrument),
  - the student achievement scores in reading and mathematics,
  - the student mathematics weights and a country adjustment factor for the reading weights and Fay's replicate,
  - The reading Fay's replicate for the computation of the sampling variance estimates.

- Intstui_scie_0701.txt consists of :
  - the student context variables collected through the student questionnaires,
  - the student composites derived from the original student context variables and the two international options (i.e. CCC and IT instrument),
  - the student achievement scores in reading and science,
  - the student science weights and a country adjustment factor for the reading weights and Fay's replicate,
  - The reading Fay's replicate for the computation of the sampling variance estimates.

- Intschoi_0701.txt consists of:
  - the school context variables collected through the school questionnaire,
  - the school composites derived from the original school context variables,
  - the school weight.

- Intconqrd_0701.txt consists of the data collected through the assessment booklets. Each item is recoded in a one-digit format. For two digit items, this file contains a single digit which is the score on the item.
- Intmatdd_0701.txt and Intscidd_0701.txt respectively consist of the mathematics and science items in a double digit-format so that double digit marked items are provided in their original format.

RECORDS INCLUDED IN THE INTERNATIONAL DATA BASE

5. At the student level, all PISA students who attended one of the cognitive sessions are included in the international database. PISA students who only attended the questionnaire

session are included if they provided an answer for the father's occupation questions or the mother's occupation questions on the student questionnaire.

6. This means that all PISA students reported in the student tracking form as not eligible, not at school any more, excluded for physical, mental or linguistic reasons or absent are not included in the international database. Students who refused to participate in any assessment sessions are not included in the international database.

7. Finally, at the student level, all non-PISA students — that means those students who were assessed in a few countries for a national or international grade sample option — are excluded from the international database. Countries, which submitted such data to the consortium, will have that data returned separately. Records from schools where less than 25 percent of the sampled eligible students participated in the testing sessions are not included in the database.

8. At the school level, all participating schools — that is, any school for which at least 25 percents of the sampled eligible student were assessed — have a record in the international database, regardless of whether the school returned the school questionnaire.

MISSING DATA

9. Four different codes are provided for the coding of missing data:

- Missing codes: 9 for a one-digit variable, 99 for a two-digit variable. For longer numerical variables, the missing codes are given in the codebook. This missing code is used only if the student had to answer but did not put any answer on the questionnaire.
- Multiple responses: 8 for a one-digit variable, 98 for a two-digit variable. This code is used for multiple choice items in both cognitive booklets and questionnaires. This code is not used for open-ended questions.
- Not applicable: 7 for a one-digit variable, 97 for two digit variables, 997 for three-digit variables for the student questionnaire data file and for the school data file. Code "n" is used for a one-digit variable in the three files that contains cognitive data. If a cognitive variable consists of more than one digit, then the same codes as for the questionnaire data files are used. For variables longer than 1 digit, the code assigned to not-applicable is equal to missing code minus two. This code was used when it was not possible for the student to answer the question. For instance, this code is used if a question has been misprinted or if a question has been deleted from the questionnaire. Not-applicable code and especially code "n" are also used in the cognitive file to fill in questions that were not included in the booklet the student received.
- Not reached items: all consecutive missing values starting from the end of each cognitive session were replaced by non-reached code, except the first value of the missing series. *Non-reached items* were considered as *not-administered* for the item calibration but were considered as incorrect answers for the estimation of the population achievement. *Non-reached* items for students who were present partly were recoded as *not applicable*. Therefore, the student is not penalised for not attending the whole session.

THE STUDENT AND SCHOOL IDENTIFICATION

10. The full student identification contained in the six student files consists of three variables:

- The country identification variable labelled COUNTRY,
- The school identification variable labelled SCHOOLID and,
- The student identification variable labelled STIDSTD.

11. In the earlier release of the data base four variables were used to identify each student. The former variables STIDSTRT and STIDSCH of 2 and 3 characters length were concatenated

to SCHOOLID of 5 characters length, which now uniquely identifies each school. The two former variables can be read by using the appropriate syntax file since no changes to the data, i.e. the 14 character string sequence, that fully identifies each student, have been introduced.

12. A fifth variable has been included to differentiate sub-national entities within countries. This variable (SUBNATIO) is meaningful for three countries only.

13. The school identification contained in the school file consists of the first two variables of the student identification, i.e. COUNTRY and SCHOOLID. Again, the two variables used formerly can be read from the data using the appropriate syntax.

THE STUDENT QUESTIONNAIRE FILE

THE OBSERVED VARIABLES COLLECTED THROUGH THE STUDENT QUESTIONNAIRES.

14. The student files contain the original variables collected through the student context questionnaires, i.e. the compulsory student questionnaire and the two international options, the Cross-Curriculum Competencies questionnaire (or CCC) and the Information Technologies questionnaire or IT.

15. The variable names used in the international database directly refer to the international version of the context questionnaires provided in Appendix 1. Each variable name consists of seven characters.

- The first two characters refer to the instrument:
  - ST for the student questionnaire,
  - IT for the Information Technologies questionnaire,
  - CC for the Cross Curriculum Competencies questionnaire.

- The third and fourth characters refer to the question number as it is in the international version of the questionnaire. For instance, ST01 refers to the first question of the student questionnaire related to the date of birth.
- The fifth character is always the letter "Q", whatever the instrument.
- The sixth and seventh characters refer to the item number of the question. For instance, ST01Q01 is the day of birth, ST01Q02 is the month of birth and ST01Q03 is the year of birth.

THE STUDENT WEIGHTS AND REPLICATES

The reading, mathematics and science weights

16. Students included in the final PISA sample for a given country are not equally representative of the full student population. To account for this in the analysis, so as not to produce biased results, surveys weights must be incorporated into the analysis.

17. The weight, W_{ij}, for student *j* in school *i* can be expressed in the following form:

$$W_{ij} = f_{1i} f_{2i} f_{1i}^{A} w_{2ij} w_{1i}, \text{ where}$$

- w_{1i} is given as the reciprocal of the probability of inclusion of school *i* into the sample;
- w_{2ij} is given as the reciprocal of the probability of selection of student *j* from within the selected school *i*;

- f_{1i} is an adjustment factor to compensate for non-participation by other schools that are somewhat similar in nature to school i (not already compensated for by the participation of replacement schools);
- f_{1i}^{A} is an adjustment factor to compensate for the fact that, in some countries, in some schools only 15-year-old students who are enroled in the modal grade for 15-year-olds were included in the assessment; and
- f_{2i} is an adjustment factor to compensate for the absence of achievement scale scores from some sampled students within school *i* (who were not excluded).

18. In the international database, the variable called W_FSTUWT is the final student weight. The sum of the weights, constitutes an estimate of the size of the target population, i.e. the number of 15 year-old students attending a school.

19. The weights described above are appropriate for the analysis of data that has been collected from all assessed students, such as student questionnaire data, and reading achievement data. Because a special booklet (Book 0) was used in some countries, assigned to certain kinds of students, additional weighting factors are required for analyses of data that are obtained from only a subset of the 10 PISA test books.

20. The mathematics weight factor is given as:

- 1.0 for each student assigned Book 0;
- 1.8 for each student assigned Book 1, 3, 5, 8, or 9;
- 0.0 for each student assigned Book 2, 4, 6, or 7 (as these books contain no mathematics test items).

21. The science weight factor is given as:

- 1.0 for each student assigned Book 0;
- 1.8 for each student assigned Book 2, 4, 6, 8, or 9;
- 0.0 for each student assigned Book 1, 3, 5, or 7 (as these books contain no science test items).

22. The mathematics weights or the science weights included respectively in "intstui_math_0701.txt" and "intstui_scie_0701.txt" were computed by multiply the reading weight, W_FSTUWT, by the respective adjustment factor. It is worth noting that the weights provided in the student mathematics file and in the student science file have already been adjusted so that no transformation are required any more.

23. A third adjustment factor, not provided in the current database, is needed to analyse the relationship between mathematics and science achievement or to perform a canonical regression with the three domain achievement data as dependent variables. Due to the remaining sample size, i.e. two ninths of the reading sample size, the consortium does not encourage an extensive use of this type of analyses.

24. The mathematics and science adjustment factors are included in the reading ASCII files (intstui_read_0701) but are not defined in the SAS or SPSS control file. These two factors are from column 788 to column 796 for mathematic and from column 797 to column 805 for science. In one want to compute the correlation between mathematics and science or to perform multivariate analyses with these two domains, these two adjustment factors should be firstly read and then the following syntax should be applied.

25. The syntax to compute this third adjustment factor is presented hereafter.

```
Compute W_MSFAC=o.
If (W_MFAC=1.8 and W_SFAC=1.8) W_MSFAC=4.5.
```

```
If (W_MFAC=1.0 and W_SFAC=1.0) W_MSFAC=1.
Execute.
```

The Fay's replicates

26. To estimate the sampling variances of PISA estimates, a replication methodology has to be employed. This will reflect the variance in estimates due to the sampling of schools and students. Additional error due to the use of plausible values will be captured separately, although computationally the two components will no doubt be carried out in a single program.

27. The approach used for calculating sampling variances is known as Balanced Repeated Replication (BRR), or Balanced Half-Samples. A particular variant, known as Fay's method, has been used. This method is very similar in nature to the jackknife method that has been used in past international studies of educational achievement, such as TIMSS.

28. The variance estimator is:

$$V_{BRR}\left(X^*\right)=\frac{1}{T\left(1-K\right)^2}\sum_{t=1}^{T}\left\{\left(X_t^*-X^*\right)^2\right\}.$$

29. Where X^* is the estimate of a given statistic from the full sample, X_t^* a set of *T* replicate estimates and *K* the Fay's coefficient. For PISA 2000, 80 replicates were computed and the Fay's coefficient was set to $K=0.5$. Therefore, the factor $\frac{1}{T\left(1-K\right)^2}$ is equal to $\frac{1}{20}$.

30. The three student questionnaire data files contain the 80 replicates that should be used to estimate the sampling variances for the computed statistics. These 80 replicates are called W_FSTR1 to W_FSTR80. These 80 replicates should only be used for analysing the appropriate achievement data and for the questionnaire data.

31. The Fay's replicates included in the mathematics and science files were transformed with the adjustment factor. The syntax used for this transformation is presented hereafter. Again, the transformations were already introduced in the mathematics and science files so that the data can be used without any additional transformations.

```
Define Matrep().
Vector Mrepli(80).
!Do !i=1 !to 80.
Compute !concat('Mrepli',!i)=!concat('w_fstr',!i) * w_mfac.
!Doend.
!Enddefine.
Matrep.

Define Scirep().
Vector Srepli(80).
!Do !i=1 !to 80.
Compute !concat('Srepli',!i)=!concat('w_fstr',!i) * w_sfac.
!Doend.
!Enddefine.
Scirep.
```

32. When importing the data into Wesvar, these 80 replicates should be allocated into the windows with the heading "replicates". The method used to create the replicates has also to be specified. It is of prime importance that the user selects the Fay's method and sets the Fay coefficient (Fay_K) as 0.5. If one does not select the method used for the replicate computation, the software will provide biased estimates of the sampling variance.

33. It is advisable to prepare three WesVar files to analyse the data. The first one will contain the data from the intstui_read_0701.txt data file and should be used for analyses of the reading achievement data and questionnaire data. The second one will contain the data from intstui_math_0701.txt (with weight W_FSTUWT and the 80 replicates adjusted for mathematics) and should be used for analyses of the mathematics achievement data. The third one will be devoted for science analyses and should be read from intstui_scie_0701.txt (with weight and replicates adjusted for the science data).

34. If someone wants to write their own program, using the replicate weights, then they should proceed as follows:

- Step 1: Calculate each estimate of interest 81 times - once with the final student weights, and once with each of the replicate weights.
- Step 2: Calculate the sum of the 80 squared differences between each of the replicate estimates in turn and the "full sample" estimate.
- Step 3: Divide the result by 20 to get the variance. This factor is constant no matter which PISA data one is analysing - formally it is dividing by 80 (for the 80 replicates) and multiplying by 4 (because of the use of Fay's method with K = 0.5 since $\frac{1}{(1-0.5)^2} = 4$).
- Step 4: Take the square root to get the standard error. The result should be identical to the result from WesVar.

Country weight adjustment factors

35. Each of the three student files contains a country adjustment factor respectively called CNTRFAC, CNTMFAC and CNTSFAC for the reading, mathematics and science files.

36. These adjustment factors are designed to equalize the sum of the weights per country to a constant set at 1000. The adjustment factors should be applied on the student weights and on the Fay's replicate.

37. The following syntax can be used to compute these country weights for the reading file.

```
Compute cntwgt=w_fstuwt*cntrfac.
Define cntrrep().
Vector Rrepli(80).
!Do !i=1 !to 80.
Compute !concat('Rrepli',!i)=!concat('w_fstr',!i) * cntrfac.
!Doend.
!Enddefine.
cntrrep.
```

THE STUDENT ACHIEVEMENT SCORES

Achievement scores

38. For each domain, i.e. reading, mathematics and science, and for each subscale in reading, two kinds of estimates are provided: a Weighted Likelihood Estimate and Plausible Values.

39. Weighted Likelihood Estimates are weighted Maximum Likelihood Estimates and should be used if the student ability estimates have to be returned to the schools. Plausible values should not be provided to schools, as plausible values are not test scores. Maximum Likelihood Estimates constitute the best estimate of a particular student's performance.

40. As described below, it is recommended to use the Plausible Values when analysing and reporting statistics at the population level.

41. The international database counts six Weighted Likelihood Estimates, respectively labelled:

- WLEMATH for the mathematics ability,
- WLEREAD for the reading ability,
- WLEREAD1 for the reading ability, *retrieving information* subscale,
- WLEREAD2 for the reading ability, *interpreting* subscale,
- WLEREAD3 for the reading ability, *reflecting* subscale,
- WLESCIE for the science ability.

42. The international database counts five plausible values for each domain and each reading subscale, respectively labelled:

- PV1MATH to PV5MATH for the mathematics ability,
- PV1READ to PV5READ for the reading ability,
- PV1READ1 to PV5READ1 for the reading ability, *retrieving information* subscale,
- PV1READ2 to PV5READ2 for the reading ability, *interpreting* subscale,
- PV1READ3 to PV5READ3 for the reading ability, *reflecting* subscale,
- PV1SCIE to PV5SCIE for the science ability.

43. Each student included in the international database has achievement scores for the reading domain as well as the reading subscales. Only students who were assessed with booklets 0, 1, 3, 5, 8, 9 will have achievement scores in mathematics and only students who were assessed with booklets 0, 2, 4, 6, 8, 9 will have achievement scores in science.

Transformation of the ability estimates to a 500 100 scale

44. The Weighted Likelihood estimates and the Plausible Values were transformed to a scale with a mean of 500 and a standard deviation of 100 by using the data of the participating OECD countries only[1].. These linear transformations used weighted data but the weights provided in the international database were transformed so that the sum of the weights per country was a constant.

45. The Weighted Likelihood Estimates for Mathematics, Reading and Science have an international average of 500 and a standard deviation of 100. The transformation that was derived to give reading a mean of 500 and standard deviation of 100 was also applied to the reading sub-scales. This means that the mean and the standard deviation for the reading sub-scales will not me 500 and 100 respectively.

46. The linear transformation applied to the Plausible Values is based on the same rules as the ones used for the Weighted Estimates, except that the standardisation parameters was derived from the average of the mean and standard deviation computed from the five PVs variables. This means that the mean and variance of the individual plausible values will not be exactly 500 and 100 respectively. But, the average of the five means and the five variances will be 500 and 100.

The Analysis of Data with Plausible Values

47. It is very important to recognise that plausible values are *not* test scores and should not be treated as such. Plausible values are random numbers that are drawn from the distribution of scores that could be reasonably assigned to each individual. As such the plausible values contain random error variance components and are not optimal as scores for individuals. The beauty of plausible values is that as a set they are better suited to describing the performance of the population than is a set of scores that are optimal at the individual student level.

[1] With the exception The Netherlands which did not reach the PISA 2000 sampling standards.

48. Plausible values are intermediate values that are provided so that consistent estimates of population parameters can be obtained using standard statistical analysis software such as SPSS and SAS.

49. During the exploring data phase, there is no need to work with the five plausible values. On the other hand, when reporting the final results, it is recommended to summarise the information contained in the five plausible values. This means that (ideally) the analysis should be undertaken five times, once with each of the five relevant plausible values variables. The results of these five analyses are averaged and then significance tests that adjust for variation between the five sets of results are computed.

50. Let us suppose that one is interested in the correlation between the student reading ability, denoted X, and a context variable Y, collected through the student questionnaire. The correlation between X and Y, denoted $r^*(X,Y)$ should be computed for each of the five plausible values. The correlation that has to be reported will be the average of the five computed correlations:

$$r^*(X,Y) = \frac{1}{5}\sum_{i=1}^{5} \hat{r}_m ,$$

where $\hat{r}_m$ is the estimate of *r* computed using the *m*-th set of plausible values.

51. We can see that the final estimate of *r* is the average of the estimates computed using each plausible value in turn. If U_m is the sampling variance for $\hat{r}_m$ then the sampling variance of r^* is:

$$V = U^* + \left(1 + M^{-1}\right) B_M ,$$

where $U^* = \frac{1}{M}\sum_{m=1}^{M} U_m$ and $B_M = \frac{1}{M-1}\sum_{m=1}^{M}\left(\hat{r}_m - r^*\right)^2$.

52. An α-% confidence interval for r^* is $r^* \pm t_\upsilon\left(\left(1-\alpha\right)\big/2\right)V^{1/2}$ where $t_\upsilon(s)$ is the s percentile of the *t*-distribution with ν degrees of freedom. $\upsilon = \frac{1}{\frac{f_M^2}{M-1} + \frac{\left(1-f_M\right)^2}{d}}$, $f_M = \left(1+M^{-1}\right)B_M \big/ V$ and *d* is the degrees of freedom that would have applied if θ_n had been observed. In PISA the value of *d* will be equal to 80.

53. It is worth noting that WesVar can analyse the plausible values for some statistics like means scores. The standard error returned by WesVar therefore includes the measurement error and the sampling error. Nevertheless, WesVar does not compute the degree of freedom as described above but just uses *d* instead, which is quite a good approximation most of the time. The current version of Wesvar can not analyse simultaneously plausible values for statistics like medians, other percentiles, percent above score cutpoints or regression analyses.

Described proficiency scale levels

54. The cutpoints for the reading scales and sub-scales are 334.75, 407.67, 480.18, 552.89 and 625.61. The five levels are defined as follow:

- Level 0: the reading score is equal or below 334.75;
- Level 1: the reading score is greater than 334.75 and equal or below 407.67;
- Level 2: the reading score is greater than 407.67 and equal or below 480.18;
- Level 3: the reading score is greater than 480.18 and equal or below 552.89;
- Level 4: the reading score is greater than 552.89 and equal or below 625.61;
- Level 5: the reading score is greater than 625.61.

55. To estimate the percentages of students in each of the six levels, five new categorical variables should be computed, each of them derived from one of the five plausible values provided by scale or subscales.

56. The following SPSS syntax might be used to derive these categorical variables.

```
Compute pv1Rlev=0.
If (PV1read > 334.75 and PV1read <=407.67) pv1Rlev=1.
If (PV1read > 407.67 and PV1read <=480.18) pv1Rlev=2.
If (PV1read > 480.18 and PV1read <=552.89) pv1Rlev=3.
If (PV1read > 552.89 and PV1read <=625.61) pv1Rlev=4.
If (PV1read > 625.61) pv1Rlev=5.
```

57. The same computation should be applied for PV2READ, PV3READ, PV4READ and PV5READ.

58. Percentages and sampling variances can be estimated with Wesvar for each of these categorical variables. The results then need to be combined as described in the previous section.

THE STUDENT COMPOSITES

59. Two kinds of composites are provided in the student questionnaire files. The first set of composites are based on a transformation of one variable or on a combination of the information included in two or more variables. The second set of composites is the result of a Rasch scaling and consists of Weighted Likelihood Estimates. The latter was preferred to factorial scores to better deal with missing data. Indeed, if a student has not answered one or more of the items, no factorial scores can be returned for this student, unless imputation methods are used. On the other hand, Item Response Theory models can deal with incomplete design so that each student will have a score, unless this student did not answer any item that composes the scale.

60. The Weighted Estimate composites were computed in three steps:

- Firstly, the item parameters were estimated on a subsample of 500 students from all OECD countries. Non-OECD countries did not contribute to the computation of the item parameters.
- The Weighted Estimates are then computed for all students and all scales; the item parameters computed previously are fixed.
- Finally, the Weighted Estimate scales are standardised with a mean of 0 and a standard deviation of 1. The statistics used for this linear transformation are based on the data of the OECD countries (except Netherlands). Like the linear transformations used for the standardisation of the achievement estimates, the weights were transformed so that the sum of the weights per country is a constant. Nevertheless, the linear transformation was applied to all countries.

Composites directly derived from the original data

61. Eleven composites were included in the student questionnaire file. The SPSS syntax to compute these eleven composites is included in Appendix 2 These are:

- Three composites that give the time in terms of minutes spent each week at school in the three assessed domains. These three variables were labelled RMINS for reading courses, MMINS for mathematics courses and SMINS for the science courses. The three variables are simply the product of the corresponding item of the student question 27 (ST27Q01, ST27Q03, ST27Q05) and the school question 6 (SC06Q03).
- The variable AGE, which gives the age of the student expressed in months.
- The structure of the family or FAMSTRUC is based on the first four items of the student question 4 (ST04Q01, ST04Q02, ST04Q03 and ST04Q04). The meaning of the four categories are : 1 is a single family, 2 is a nuclear family, 3 is a mixed family and 4 groups the other responses, except the non-responses which are maintained as missing or not applicable.
- NSIB or the number of siblings is based on the three items of question ST25Q01.
- BRTHORD or the birth order of the assessed student. It will be equal to 0 is the student is the only child, 1 if the student is the youngest child, 2 is the student is the middle child and 3 if the student is the oldest child.
- The mother's occupation, the father's occupation and the student expected occupation at the age of 30 respectively observed through questions ST08Q01, ST09Q01, ST10Q01 ST11Q01 and ST40Q01 were coded into the International Standard Classification of Occupation. These ISCO categories were then transformed into an International Socio-Economic Index according to the methodology developed by Ganzeboom, de Graaf and Treiman (1992). The three following variables (BMMJ, BFMJ and BTHR) are respectively the mother's ISEI, father's ISEI and student self expected ISEI. Two new derived variables were computed from BMMJ and BFMJ. ISEI is equal to BFMJ, i.e. the father ISEI or to BMMJ if the father's ISEI is missing. HISEI corresponds to the highest value of BMMJ and BFMJ.
- The educational level of the father and the mother were collected by two questions for each parent (questions ST12q01 and ST14Q01 for the mother and questions ST13Q01 and ST15Q01 for the father). The father educational level (FISCED) and the mother educational level (MISCED) have the following categories:
  - 1 did not go to school,
  - 2 ISCED Level 1 only (primary education),
  - 3 ISCED Level 2 only (Lower secondary level)
  - 4 ISCED Level 3B or 3C only (Upper secondary level)
  - 5 ISCED Level 3A (Upper secondary)
  - 6 ISCED Level 5A, 5B or 6 (Tertiary education)

Weighted Likelihood Estimate Composites

Student background questionnaire

62. Fifteen IRT scale composites were derived from the student questionnaire. These composites are:

- CULTCOM, derived from questions ST19Q01, ST19Q02 and ST19Q03.
- SOCCOM, derived from questions ST19Q04, ST19Q05 and ST19Q06.
- FAMEDSUP, derived from questions ST20Q01, ST20Q02 and ST20Q03.
- WEALTH, derived from questions STD21Q01, STD21Q02, STD21Q03, STD21Q04, ST22Q01, ST22Q02, ST22Q04, ST22Q06 and ST22Q07.
- HEDRES, derived from questions STD21Q05, STD21Q06, STD21Q07, STD21Q08, ST22Q03.
- CULTACTV, derived from questions ST18Q02, ST18Q04 and ST18Q05.
- CULTPOSS, derived from questions ST21Q09, ST21Q10 and ST21Q11.
- HMWKTIME, derived from questions ST33Q01, ST33Q02 and ST33Q03.

- TEACHSUP, derived from questions ST26Q05, ST26Q06, ST26Q07, ST26Q08, ST26Q09 and ST26Q10.
- DISCLIMA, derived from questions ST26Q01, ST26Q12, ST26Q13, ST26Q14, ST26Q16 and ST26Q17.
- STUDREL, derived from questions ST30Q01- TO ST30Q05.
- ACHPRESS, derived from questions ST26Q02, ST26Q03, ST26Q04 and ST26Q15.
- BELONG, derived from questions ST31Q01 TO ST31Q06.
- JOYREAD, derived from questions ST35Q01 TO ST35Q09.
- DIVREAD, derived from questions ST36Q01 TO ST36Q06. For these last composites, categories 1 and 2 were recoded as 0 and categories 3, 4,5 were recoded as 1.

63. The IRT composites were transformed to a mean of 0 and a standard deviation of 1 at the international level using the same procedures as were applied to the achievement variables.

IT composites

- COMAB derived from variables IT02Q01 IT02Q02 IT02Q03 IT03Q01 (items inverted).
- COMUSE derived from variables IT05Q03 IT05Q04 IT06Q02 IT06Q03 IT06Q04 IT06Q05 (items inverted).
- COMATT derived from variables IT07Q01 IT08Q01 IT09Q01 IT10Q01 (items inverted).

64. The IRT composites were transformed to a mean of 0 and a standard deviation of 1 at the international level using the same procedures as were applied to the achievement variables. Only country that participate to the IT familiarity questionnaire were included in this transformation.

CCC COMPOSITES:

- CTRAT derived from variables CC01Q03, CC01Q13, CC01Q19, CC01Q23 and CC01Q27.
- EFFPER derived from variables CC01Q07, CC01Q12, CC01Q20 and CC01Q28.
- MEMOR derived from variables CC01Q01, CC01Q05, CC01Q10 and CC01Q15.
- SELFEF derived from variables CC01Q02, CC01Q18 and CC01Q26.
- CEXP derived from variables CC01Q04, CC01Q11, CC01Q16 and CC01Q24.
- ELAB derived from variables CC01Q09, CC01Q17, CC01Q21 and CC01Q25.
- INSMOT derived from variables CC01Q06, CC01Q14 and CC01Q22.
- INTMAT derived from variables CC02Q01, CC02Q10 and CC02Q21.
- MATCON derived from variables CC02Q12, CC02Q15 and CC02Q18.
- INTREA derived from variables CC02Q06, CC02Q13 and CC02Q17.
- SCACAD derived from variables CC02Q03, CC02Q07 and CC02Q20.
- SCVERB derived from variables CC02Q05, CC02Q09 and CC02Q23.
- COMLRN derived from variables CC02Q04, CC02Q11, CC02Q16 and CC02Q24.
- COPLRN derived from variables CC02Q02, CC02Q08, CC02Q19 and CC02Q22.

65. The IRT composites were transformed to a mean of 0 and a standard deviation of 1 at the international level using the same procedures as were applied to the achievement variables. Only country that participate to the CCC were included in this transformation. Nevertheless, Scotland participated while England and North Ireland dit not. Therefore, Scotland was not included in the countries that contribute to the standardisation.

THE SCHOOL FILE

THE OBSERVED VARIABLES COLLECTED THROUGH THE SCHOOL QUESTIONNAIRE.

66. The school files contain the original variables collected through the school context questionnaire.

67. The variable names used in the international database directly refer to the international version of the context questionnaire provided in Appendix 1. Each variable name consists of seven characters.

- The first two characters refer to the instrument: SC
- The third and fourth characters refer to the question number as it is in the international version of the questionnaire. For instance, SC02 refers to the second question of the school questionnaire related the total enrolment.
- The fifth character is always the letter "Q".
- The sixth and seventh characters refer to the item number of the question. For instance, SC02Q01 is the total number of boys and SC02Q02 is the total number of girls enroled in the school.

THE SCHOOL WEIGHT

68. The school base weight adjusted for non-response is provided at the end of the school file. No replicates are provided in the school file.

THE SCHOOL COMPOSITES

69. As in the student questionnaire data file, two kinds of composites were derived from the school questionnaire data.

Composites directly derived from the original data

70. The derived composites are mainly related to the school size, the computer environment of the school and the school staffing. The SPSS syntax used to compute these composites is presented in Appendix 3.

- SCHLSIZE is the sum of question SC02Q01 and SC02Q02.
- PCGIRLS is the ratio between the number of girls and the total enrolment, i.e. the school size.
- SCHLTYPE is based on the question SC03Q01 and SC04Q01 to SC04Q04. This variable has the following categories:
  - 1 Private, Government independent,
  - 2 Private, Government dependent,
  - 3 Government.

- TOTHRS consists of the total number of 60 minute schooling hours per year and is the product of question SC06Q01, SC06Q02 and SC06Q03 divided by 60.
- RATCOMP is the ratio between the total number of computers in the school divided by the school size.
- PERCOMP1 is the ratio of the number of computers available for 15 year-olds (sc13Q02) divided by the total number of computers (SC13Q01).
- PERCOMP2 is the ratio of the number of computers available for teachers only (sc13Q03) divided by the total number of computers (SC13Q01).
- PERCOMP3 is the ratio of the number of computers available only for administrative staff (sc13Q04) divided by the total number of computers (SC13Q01).

- PERCOMP4 is the ratio of the number of computers connected to the Web (sc13Q05) divided by the total number of computers (SC13Q01).
- PERCOMP5 is the ratio of the number of computers connected to a local network (sc13Q06) divided by the total number of computers (SC13Q01).
- STRATIO is the ratio between the school size and the total number of teachers. Part-time teachers contributed 0.5 and full-time teachers 1.0 to the total number of teachers. This rule also applies to all composites based on question SC14.
- PROPQUAL is the ratio between the total number of teachers who have an ISCED 5A qualification in pedagogy and the total number of teachers.
- PROPCERT is the ratio between the total number of teachers who are fully certified by the appropriate authority and the total number of teachers.
- PROPREAD is the ratio between the test language teachers who have an ISCED 5A qualification and the total number of test language teachers.
- PROPMATH is the ratio between the mathematics teachers who have an ISCED 5A qualification and the total number of mathematics teachers.
- PROPSCIE is the ratio between the science teachers who have an ISCED 5A qualification and the total number of science teachers.

Weighted Likelihood Estimate Composites

71. Eight IRT scale composites were derived from the school questionnaires. These composites are:

- SCMATEDU, derived from questions SC11Q04 TO SC11Q09.
- SCMATBUI, derived from questions SC11Q01 TO SC11Q03.
- TCSHORT, derived from questions SC21Q01 TO SC21Q04.
- TEACHBEHA, derived from questions SC19Q01, SC19Q03, SC19Q07, SC19Q08, SC19Q11, SC19Q14 and SC19Q16.
- STUDBEHA, derived from questions SC19Q02, SC19Q06, SC19Q09, SC19Q10, SC19Q13 and SC19Q15.
- TCMORALE, derived from questions SC20Q01 to SC20Q04.
- SCHAUTON, derived from questions SC22Q01 to SC22Q12 but use only the data for the first proposed category, i.e. “Not a school responsibility”. Items were inverted.
- TCHPARTI, derived from questions SC22Q01 to SC22Q12 but use only the data for the first proposed category, i.e. “Teachers”.

72. The IRT composites were transformed to a mean of 0 and a standard deviation of 1 at the international level using the same procedures as were applied to the achievement variables, i.e. each OECD country, except Netherlands, contributes equally to the standardisation.

THE COGNITIVE FILES

73. As stated earlier, three cognitive files are proposed in the international database. The biggest one, INTCONQRD.TXT, contains all items used for the international item calibration and computation of the Plausible Values. All items included in this file have a one-digit format. The two other files, called INTMATDD.TXT and INTSCIDD.txt respectively consist of the mathematics items and the science items in a two-digit format.

ONE DIGIT FORMAT COGNITIVE FILE

Recoding of the cognitive items

74. The PISA items are organised into units. Each unit consists of one or more questions. Each unit is identified by a short label and by a long label. All units’ short labels consist of four characters. The first character is R, M or S respectively for Reading, Mathematics or

Science. The three next characters are three digit numbers. The long label is related to the question content. The items within a unit have the same long label and have in most cases a short label of seven characters, which include the four-character label of the unit plus a question number such as Q03. For instance, the two labels of a mathematics item are M159Q01 and Speed of Racing car. All items associated to this unit have a short label starting by M159.

75. In the cognitive files, the items are sorted by domain and within domain by alphabetical order according to the short label. It means that the mathematics items appear first, then the reading items and the science items. Within a domain, the items of unit M037 appear before the items of unit M136. Within a particular unit, the item labelled Q01 will appear before the item labelled Q02.

76. Some of the items needed to be recoded before being used for the national and international scaling. The SPSS syntax for the recoding in included in appendix 4.

- Double-digit items were truncated by retaining only the first digit, which corresponds to the score initially assigned to the item. The list of double-digit items and the initial and recoded categories are presented in exhibit 1.

Exhibit 1: Initial and recoded categories of the double-digit item

Short Label	*Initial codes*	*Recoded codes*
M124Q03	00;11;21;22;23;24;31;99;97	0, 1, 2, 3, 9, n
M136Q01	01;02;11;12;21;99;97	0, 1, 2, 9, n
M136Q02	00;11;12;13;14;15;99;97	0, 1, 9, n
M136Q03	01;02;11;21;97;99	0, 1, 2, 9, n
M148Q02	01;02;11;12;13;14;21;22;23;24;25	0, 1, 2, 9, n
M150Q02	00;11;21;22;99;97	0, 1, 2, 9, n
M150Q03	01;02;11;12;13;99;97	0, 1, 9, n
M155Q02	00;11;12;13;21;99;97	0, 1, 2, 9, n
M155Q03	00;11;12;13;21;22;23;99;97	0, 1, 2, 9, n
M179Q01	01;02;03;04;11;12;21;22;23;99;97	0, 1, 2, 9, n
S114Q03	01;02;11;12;99;97	0, 1, 9, n
S114Q04	01;02;03;11;12;13;14;15;21;99;97	0, 1, 2, 9, n
S114Q05	01;02;03;11;12;99;97	0, 1, 9, n
S129Q02	01;02;03;04;11;12;13;21;99;97	0, 1, 2, 9, n
S131Q02	01;02;03;11;12;99;97	0, 1, 9, n
S131Q04	01;02;03;04;11;99;97	0, 1, 9, n
S195Q02	01;02;03;04;11;12;13;21;99;97	0, 1, 2, 9, n
S195Q05	01;02;11;12;13;14;15;99;97	0, 1, 9, n
S209Q02	01;02;03;11;12;13;14;15;99;97	0, 1, 9, n
S253Q01	01;11;12;13;21;22;23;31;99;97	0, 1, 2, 3, 9, n
S268Q02	01;02;03;11;12;13;14;15;99;97	0, 1, 9, n
S269Q03	01;02;11;12;99;97	0, 1, 9, n

- Other items were recoded and/or combined. These items have been re-labelled. The character "T" was added to the end of the previous short label.
  - Numerical variables were recoded into scores, i.e. incorrect answer (0), correct answer (1), missing answer (9) or Not applicable (7). Exhibit 2 presents the list of the numerical items, the data entry validation rules and the codes after recoding.

Exhibit 2: Recoding of the numeric items

Short Label	*Initial codes*	*Recoded codes*
M034Q01	Between 0 and 90 or 99 or 97	0, 1, 9, n
M037Q01	Between 0 and 900 or 999 or 997	0, 1, 9, n
M037Q02	Between 0 and 90 or 99 or 97	0, 1, 9, n
M144Q01	Between 0 and 90 or 99 or 97	0, 1, 9, n
M144Q02	Between 0 and 90 or 99 or 97	0, 1, 9, n
M144Q04	Between 0 and 90 or 99 or 97	0, 1, 9, n

  - Some questions consisted of several true/false or yes/no items. Two questions were also composed of several multiple choice type items (R088Q04 and R099Q03). These items were combined into a new variable. The new codes correspond to the number of correct answers in the subset of items.

Exhibit 3: Recodding of the complex multiple choice items

Short Label	*Initial codes*	*Number of subsets*	*New codes*
M155Q04	1;2;8;9;n	4	0, 1, 2, 3, 4, 8, 9
M266Q01	1;2;8;9;n	4	0, 1, 2, 3, 4, 8, 9
R070Q07	1;2;8;9;n	5	0, 1, 2, 3, 4, 5, 8, 9
R088Q04	1;2;3;4;8;9;n	5	0, 1, 2, 3, 4, 5, 8, 9
R088Q05	1;2;8;9;n	3	0, 1, 2, 3, 8, 9
R099Q03	1;2;3;8;9;n	3	0, 1, 2, 3, 8, 9
R120Q07	1;2;8;9;n	3	0, 1, 2, 3, 8, 9
R122Q03	1;2;8;9;n	6	0, 1, 2, 3, 4, 5, 6, 8, 9
R227Q02	1;2;8;9;n	7	0, 1, 2, 3, 4, 5, 6, 7, 8,
S128Q03	1;2;8;9;n	2	0, 1, 2, 8, 9
S133Q04	1;2;8;9;n	3	0, 1, 2, 3, 8, 9
S209Q01	1;2;8;9;n	4	0, 1, 2, 3, 4, 8, 9
S213Q01	1;2;8;9;n	4	0, 1, 2, 3, 4, 8, 9
S252Q03	1;2;8;9;n	2	0, 1, 2, 8, 9
S269Q04	1;2;8;9;n	4	0, 1, 2, 3, 4, 8, 9
S270Q03	1;2;8;9;n	2	0, 1, 2, 8, 9

- Finally, five items (R119Q09, R122Q01, R216Q03, R219Q01 and M192Q01), composed of a subset of items, were combined to form a new variable. The combined codes correspond to the number of correct answer to each of the sub-items included in these five items.

National and international item deletions

77. For the PISA main study, 203 items were included in the nine rotated booklets and 30 of these 203 items were included in the Special Education booklet.

78. The first step of the scaling analysis consists of a national item calibration. In most cases, all 203 items were included in this national scaling. Nevertheless, some countries detected an error before submitting the data to the consortium and in agreement with the International Coordinating Centre, these items were recoded as *Not Applicable* so that they were not included in the national scaling.

79. During the item adjudication process, it was decided to delete a few items at the international level and to consult countries to perform additional checks on dodgy items. This consultation has lead to the deletion of some items at the national level. The items deleted at the international level are: M148Q01, R081Q02, R093Q04, R099Q02, R099Q03, R122Q01, R225Q01, R241Q01 and S209Q01. These items are not included in the international database.

Exhibit 4: Items deleted for a particular country

Country	*Item name*
Austria	M155Q03
Austria	R055Q03
Austria	S133Q04T
Belgium Dutch version	R076Q05
Belgium Dutch version	R100Q05
Brazil	M033Q01
Canada French version	R101Q08
England	R076Q03
England	R076Q04
Germany	R055Q03
Germany	S133Q04T
Greece	R040Q02
Hungary	R119Q04
Iceland	R236Q01
Iceland	S268Q02T
Italy	R040Q06
Italy	R219Q01T
Japan	M155Q01
Korea	R102Q04A
Korea	R216Q02

Country	*Item name*
Korea	R237Q03
Korea	R246Q02
Mexico	R040Q02
Netherlands	R076Q05
Netherlands	R100Q05
Netherlands	S268Q02T
Poland	R099Q04B
Russian Republik	R091Q05
Spain	R227Q01
Sweden	R091Q07B
Switzerland, German version	M155Q01
Switzerland, German version	M155Q03
Switzerland, German version	M155Q04
Switzerland, German version	R055Q03
Switzerland, German version	R076Q03
Switzerland, German version	R091Q05
Switzerland, German version	R111q06B
Switzerland, German version	R239Q02
Switzerland, German version	S133Q04T
Switzerland, Italian version	S268Q06

International scores assigned to the items.

80. A few partial credit items showed bad fit statistics in most or all countries. Alternative scoring were tried and the scoring rules associated with the best fit statistics were retained for the international item calibration and the computation of plausible values.

81. The final scores allocated to the different categories are presented in Appendix 5. The categories codes are grouped according to the scores they were assigned for the final international calibration.

DOUBLE DIGIT FORMAT FILES

82. Some open-ended items in mathematics and in science were marked by using a double-digit format. The double-digit system was set up so that the initial code assigned to the item was the first digit and the conception or misconception indicated by the response was represented by the second digit.

83. As stated earlier, the double-digit items were transformed to one digit for the national and international scaling by keeping the first digit which represents the score assigned to the item.

84. Two files are provided with the double-digit information. The first one contains all mathematics items in a double-digit format. The second one contains all science items in a double-digit format.

85. The recoding rules, except the ones related to the double-digit items, and the national and international deletion described in the previous section, also apply to these files.

Appendix 1: Context questionnaires.

The international version of the survey comes as a supplement to this report.

Appendix 2, Student derived composite SPSS syntax

```
* Time spend on courses.

numeric rmins mmins smins (f4.0).
compute rmins=st27q01*sc06q03.
compute mmins=st27q03*sc06q03.
compute smins=st27q05*sc06q03.
recode rmins mmins smins (1000 thru highest=sysmis).
execute.
missing value st27q01 st27q03 st27q05 sc06q03 ().
execute.
recode rmins mmins smins (sysmis=9999).
if (st27q01=97 or sc06q03=997) rmins=9997.
if (st27q03=97 or sc06q03=997) mmins=9997.
if (st27q05=97 or sc06q03=997) smins=9997.
execute.
missing value st27q01 st27q03 st27q05 (97,98,99)/sc06q03 (997,998,999)/rmins
mmins smins (9997,9999).
execute.

* 1. Age

numeric age (f3.0).
Do if (country="1140" or country="3050" or country="3250" or country="5361").
compute age=((2000-numeric(st01q03,f4.0))*12)+(3-numeric(st01q02,f2.0)-1).
end if.
Do if (country="1313" or country="1410" or country="2100" or country="2130" or
country="2370" or country="3190" or country="5330"
or country="3350" or country="4090" or country="4362" or country="5431" or
country="6390" or country="7080"
or country="7290" or country="7312" or country="8240" or country="8150" or
country="8270" or country="9311" or country="9432").
compute age=((2000-numeric(st01q03,f4.0))*12)+(4-numeric(st01q02,f2.0)-1).
end if.
Do if (country="3230" or country="4200" or country="5610" or country="7120" or
country="9320" or country="7260").
compute age=((2000-numeric(st01q03,f4.0))*12)+(5-numeric(st01q02,f2.0)-1).
end if.
Do if (country="6380" or country="2210").
compute age=((2000-numeric(st01q03,f4.0))*12)+(7-numeric(st01q02,f2.0)-1).
end if.
Do if (country="2170" or country="6020").
compute age=((2000-numeric(st01q03,f4.0))*12)+(8-numeric(st01q02,f2.0)-1).
end if.
Do if (country="7060").
compute age=((2000-numeric(st01q03,f4.0))*12)+(10-numeric(st01q02,f2.0)-1).
end if.
if (st01q03="9999" or st01q03="9998" or st01q03="9997" or st01q02="97" or
st01q02="98" or st01q02="99") age=999.
if (st01q03="9997" and st01q02="97") age=997.
missing value
/ age (997,998,999).
execute.

* 2. Family structure.

numeric FAMSTRUC (f1.0).
missing value st04q01 st04q02 st04q03 st04q04 ().
execute.
count nbcount=st04q01 st04q02 st04q03 st04q04 (1).
do if (nbcount=1).
compute FAMSTRUC = 1.
else if (nbcount=2 and st04q01=1 and st04q03=1).
compute FAMSTRUC=2.
else if (nbcount=2 and st04q01=1 and st04q04=1).
compute FAMSTRUC=3.
else if (nbcount=2 and st04q02=1 and st04q03=1).
compute FAMSTRUC=3.
```

```
else if (nbcount=2 and st04q02=1 and st04q04=1).
compute FAMSTRUC=3.
else if ((st04q01=7) and (st04q02=7) and (st04q03=7) and (st04q04=7)).
compute FAMSTRUC=7.
else if ((st04q01=8) and (st04q02=8) and (st04q03=8) and (st04q04=8)).
compute FAMSTRUC=8.
else if ((st04q01=9) and (st04q02=9) and (st04q03=9) and (st04q04=9)).
compute FAMSTRUC=9.
else if ((st04q01=7 or st04q01=8 or st04q01=9) and (st04q02=7 or st04q02=8 or
st04q02=9) and (st04q03=7 or st04q03=8 or st04q03=9) and (st04q04=7 or
st04q04=8 or st04q04=9)).
compute FAMSTRUC=9.
else.
compute FAMSTRUC=4.
end if.
execute.
missing value st04q01 st04q02 st04q03 st04q04  FAMSTRUC (7,8,9).
execute.

*  3. Number of siblings.

COUNT
nbmis = st05q01 st05q02 st05q03  (MISSING)  .
execute.
DO IF (nbmis < 3) .
RECODE st05q01 st05q02 st05q03 (MISSING=1)  .
END IF .
EXECUTE .
numeric NSIB (f2.0).
compute NSIB= st05q01+st05q02+st05q03-3.
execute.
missing value st05q01 st05q02 st05q03  ().
execute.
recode NSIB (sysmis=99).
if (st05q01=7 or st05q02=7 or st05q03=7) nsib=97.
if (st05q01=8 and st05q02=8 and st05q03=8) nsib=98.
execute.
missing value st05q01 st05q02 st05q03 nsib (97,98,99).
execute.

* Birth order

numeric brthord (f1.0).
missing value st05q01 st05q02 st05q03  ().
execute.
if (st05q01=1 and st05q02=1 and st05q03=1) brthord=0.
if  ((st05q01=2  or  st05q01=3  or  st05q01=4  or  st05q01=5)  and  (st05q02=1))
brthord=1.
if ((st05q01=1) and (st05q02=1) and (st05q03=2 or st05q03=3 or st05q03=4 or
st05q03=5)) brthord=1.
if  ((st05q01=2  or  st05q01=3  or  st05q01=4  or  st05q01=5)  and  (st05q02=2  or
st05q02=3 or st05q02=4 or st05q02=5)) brthord=2.
if  ((st05q01=1)  and  (st05q02=2  or  st05q02=3  or  st05q02=4  or  st05q02=5))
brthord=3.
if  ((st05q01=7  or  st05q01=8  or  st05q01=9)  and  (st05q02=7  or  st05q02=8  or
st05q02=9) and (st05q03=7 or st05q03=8 or st05q03=9)) brthord=9.
if (st05q01=7 or st05q02=7 or st05q03=7) brthord=7.
if (st05q01=8 and st05q02=8 and st05q03=8) brthord=8.
if (st05q01=9 and st05q02=9 and st05q03=9) brthord=9.
execute.
missing value st05q01 st05q02 st05q03  brthord (7,8,9).
execute.


*  Socio-economic status

numeric ISEI HISEI (f2.0).
Compute isei=bfmj.
if missing(bfmj) isei=bmmj.
compute hisei=max(bmmj,bfmj).
```

```
execute.
missing value bmmj bfmj ().
execute.
do if (bmmj=97 and bfmj=97).
recode isei hisei (sysmis=97).
else if (bmmj=98 and bfmj=98).
recode isei hisei (sysmis=98).
else if (bmmj=99 and bfmj=99).
recode isei hisei (sysmis=99).
end if.
execute.
recode isei hisei (sysmis=99).
execute.
missing value bmmj bfmj isei hisei (97,98,99).
execute.


* Parental Education.

missing value st12q01 st13q01 st14q01 st15q01 ().
execute.
numeric FISCED MISCED (f1.0).
compute MISCED=st12q01.
if (st14q01=1) MISCED=6.
compute FISCED=st13q01.
if (st15q01=1) FISCED=6.
if (st12q01=7 and st14q01=7) MISCED=7.
if (st13q01=7 and st15q01=7) FISCED=7.
if (country="2210") MISCED=7.
if (country="2210") FISCED=7.
execute.
missing value st12q01 st13q01 st14q01 st15q01 misced fisced (7,8,9).
execute.
```

Appendix 3: School derived composite SPSS syntax

```
* school size.

numeric schlsize (f5.0).
compute schlsize = sc02q01+sc02q02.
if (missing(sc02q01)) schlsize=sc02q02.
if (missing(sc02q02)) schlsize=sc02q01.
execute.
missing value sc02q01 sc02q02 ().
execute.
recode schlsize (sysmis=99999).
if (sc02q01=9997 and sc02q02=9997) schlsize=99997.
recode schlsize (0=99999).
execute.
missing value sc02q01 sc02q02 (9997,9998,9999)/schlsize (99997,99999).
execute.

* Pcgirls

numeric pcgirls (f5.3).
compute pcgirls = sc02q02/schlsize.
if (missing(sc02q02) and not missing(sc02q01)) pcgirls=1.
execute.
missing value sc02q01 sc02q02 schlsize ().
execute.
recode pcgirls (sysmis=9).
if (schlsize=99997) pcgirls=7.
execute.
missing value sc02q01 sc02q02 (9997,9998,9999)/schlsize (99997,99999)/pcgirls
(7,9).
execute.

* School type

numeric schltype (f1.0).
compute #fund=0.
if (not missing(sc04q01)) #fund=#fund+sc04q01.
if (not missing(sc04q02)) #fund=#fund+sc04q02.
if (not missing(sc04q03)) #fund=#fund+sc04q03.
if (not missing(sc04q04)) #fund=#fund+sc04q04.
do if(#fund>90 and #fund<110).
if (sc03q01=1) schltype=3.
if (sc03q01=2 and sc04q01 <50) schltype=1.
if (sc03q01=2 and sc04q01 >=50) schltype=2.
end if.
execute.
missing value sc03q01 sc04q01 ().
execute.
recode schltype (sysmis=9).
if (sc03q01=7 or sc04q01=997) schltype=7.
execute.
missing value sc03q01 schltype (7,8,9) sc04q01 (997,998,999).
execute.


* Number of school hours.

numeric tothrs (f4.0).
compute tothrs=(sc06q01*sc06q02*sc06q03)/60.
execute.
missing value sc06q01 sc06q02 sc06q03 ().
execute.
if (tothrs < 100) tothrs=9999.
if (tothrs > 1700) tothrs=9999.
recode tothrs (sysmis=9999).
if (sc06q01=97 or sc06q02 = 97 or sc06q03=997) tothrs=9997.
execute.
```

```
missing value sc06q01 sc06q02 (97,98,99) / sc06q03 (997,998,999) / tothrs
(9997,9999).
execute.

* Ratio school size - teachers number

recode sc13q01 sc13q02 sc13q03 sc13q04 sc13q05 sc13q06 (997=9997) (999=9999).
execute.
numeric ratcomp(f6.3).
compute ratcomp=sc13q01/schlsize.
execute.
missing value sc13q01 ().
execute.
recode ratcomp (sysmis=999).
do if (sc13q01=9997 ) .
recode ratcomp (sysmis,999=997).
end if.
execute.
missing value sc13q01 (9997,9998,9999)/ ratcomp  (997,999).
execute.

* Computer composites

numeric percomp1 to percomp5 (f4.2).
do if (sc13q01>0).
compute percomp1=sc13q02/sc13q01.
compute percomp2=sc13q03/sc13q01.
compute percomp3=sc13q04/sc13q01.
compute percomp4=sc13q05/sc13q01.
compute percomp5=sc13q06/sc13q01.
end if.
if (percomp1>1.00) percomp1=7.
if (percomp2>1.00) percomp1=7.
if (percomp3>1.00) percomp1=7.
if (percomp4>1.00) percomp1=7.
if (percomp5>1.00) percomp1=7.
execute.
missing value sc13q01 to sc13q06 ().
execute.
recode percomp1 percomp2 percomp3 percomp4 percomp5  (sysmis=9).
if (sc13q01=9997 or sc13q02=9997) percomp1=7.
if (sc13q01=9997 or sc13q03=9997) percomp2=7.
if (sc13q01=9997 or sc13q04=9997) percomp3=7.
if (sc13q01=9997 or sc13q05=9997) percomp4=7.
if (sc13q01=9997 or sc13q06=9997) percomp5=7.
execute.
missing value  sc13q01 to sc13q06 (9997,9998,9999)/percomp1 to percomp5 (7,9).
execute.


* Teachers composites

numeric stratio (f5.2).
do if (sc14q01>0 or sc14q02>0).
compute stratio=schlsize/(sc14q01+(0.5*sc14q02)).
if (missing(sc14q01) and not missing(sc14q02)) stratio=schlsize/(0.5*sc14q02).
if (not missing(sc14q01) and missing(sc14q02)) stratio=schlsize/(sc14q01).
end if.
recode stratio (50 thru Highest=sysmis).
execute.
missing value sc14q01 sc14q02 ().
execute.
recode stratio (sysmis=99).
if (sc14q01=997 and sc14q02=997) stratio=97.
execute.
missing value sc14q01 sc14q02  (997,998,999)/stratio (97,99).
execute.

compute #q14a=sc14q01+(0.5*sc14q02).
if (missing(sc14q01)) #q14a=0.5*sc14q02.
```

```
if (missing(sc14q02)) #q14a=sc14q01.

compute #q14b=sc14q03+(0.5*sc14q04).
if (missing(sc14q03)) #q14b=0.5*sc14q04.
if (missing(sc14q04)) #q14b=sc14q03.

compute #q14c=sc14q05+(0.5*sc14q06).
if (missing(sc14q05)) #q14c=0.5*sc14q06.
if (missing(sc14q06)) #q14c=sc14q05.

compute #q14d=sc14q07+(0.5*sc14q08).
if (missing(sc14q07)) #q14d=0.5*sc14q08.
if (missing(sc14q08)) #q14d=sc14q07.

compute #q14e=sc14q09+(0.5*sc14q10).
if (missing(sc14q09)) #q14e=0.5*sc14q10.
if (missing(sc14q10)) #q14e=sc14q09.

compute #q14f=sc14q11+(0.5*sc14q12).
if (missing(sc14q11)) #q14f=0.5*sc14q12.
if (missing(sc14q12)) #q14f=sc14q11.

compute #q14g=sc14q13+(0.5*sc14q14).
if (missing(sc14q13)) #q14g=0.5*sc14q14.
if (missing(sc14q14)) #q14g=sc14q13.

compute #q14h=sc14q15+(0.5*sc14q16).
if (missing(sc14q15)) #q14h=0.5*sc14q16.
if (missing(sc14q16)) #q14h=sc14q15.

compute #q14i=sc14q17+(0.5*sc14q18).
if (missing(sc14q17)) #q14i=0.5*sc14q18.
if (missing(sc14q18)) #q14i=sc14q17.

if (#q14a>0)  propqual=#q14b/#q14a.
if (#q14a>0)  propcert=#q14c/#q14a.
if (#q14d>0)  propread=#q14e/#q14d.
if (#q14f>0)   propmath=#q14g/#q14f.
if (#q14h>0)  propscie=#q14i/#q14h.
recode   propqual   propcert   propread   propmath   propscie   (1.00000001   thru
Highest=9) (sysmis=9).
execute.
missing value sc14q01 to sc14q18 ().
if (sc14q01=997 or sc14q02=997 or sc14q03=997 or sc14q04=997) propqual=7.
if (sc14q01=997 or sc14q02=997 or sc14q05=997 or sc14q06=997) propcert=7.
if (sc14q07=997 or sc14q08=997 or sc14q09=997 or sc14q10=997) propread=7.
if (sc14q11=997 or sc14q12=997 or sc14q13=997 or sc14q14=997) propmath=7.
if (sc14q15=997 or sc16q02=997 or sc17q03=997 or sc18q04=997) propscie=7.
execute.
missing value sc14q01 to sc14q18 (997,998,999) / propqual propcert propread
propmath propscie (7,9).
execute.
```

Appendix 4: SPSS syntax for the recoding of the cognitive items.

```
* Declare new variables. String variables of length greater than 1
  are concatenated from multiple parts. Numeric width one variables
  ending in "T" are counts of matches for complex multiple choice items
  and scores for other types of multiple part items.

* Some scratch variables.

numeric  #NACHK (F1.0) #ACCUM (F1.0) #MISSNG (F1.0).
string #S1(A1) #S2(A1) #S3(A1) #works (a14).

string  M034Q01T M037Q01T M037Q02T M124Q03T M136Q01T M136Q02T M136Q03T (A1)
    /M144Q01T M144Q02T M144Q04T M145Q01T M148Q02T M150Q02T M150Q03T (A1)
    /M155Q02T M155Q03T M155Q04T M179Q01T M192Q01T M266Q01T M273Q01T (A1)
    /R070Q07T R088Q04T R088Q05T R099Q03T R119Q09T R120Q07T R122Q01T (A1)
    /R122Q03T R216Q03T R219Q01T R227Q02T                           (A1)
    /S114Q03T S114Q04T S114Q05T S128Q03T S129Q02T S131Q02T S131Q04T (A1)
    /S133Q04T S195Q02T S195Q05T S209Q01T S209Q02T S213Q01T S252Q03T (A1)
    /S253Q01T S268Q02T S269Q03T S269Q04T S270Q03T (A1).

string  M155Q04S (A4) M192Q01S (A3) M266Q01S (A4)
    /R070Q07S (A5) R088Q04S (A5) R088Q05S (A3) R099Q03S (A3) R119Q09S (A2)
    /R120Q07S (A3) R122Q01S (A3) R122Q03S (A6) R216Q03S (A3) R219Q01S (A4)
    /R227Q02S (A7)
    /S128Q03S (A2) S133Q04S (A3) S209Q01S (A4) S213Q01S (A4) S252Q03S (A2)
    /S269Q04S (A4) S270Q03S (A2).


* For multiple part questions, create concatenated string variable
  of the parts.

compute M155Q04S=concat(M155Q04a to M155Q04d).
compute M192Q01S=concat(M192Q01a to M192Q01c).
compute M266Q01S=concat(M266Q01a to M266Q01d).

compute R070Q07S=concat(R070Q07a to R070Q07e).
compute R088Q04S=concat(R088Q04a to R088Q04e).
compute R088Q05S=concat(R088Q05a to R088Q05c).
compute R099Q03S=concat(R099Q03a to R099Q03c).
compute R119Q09S=concat(R119Q09a to R119Q09b).
compute R120Q07S=concat(R120Q07a to R120Q07c).
compute R122Q01S=concat(R122Q01a to R122Q01c).
compute R122Q03S=concat(R122Q03a to R122Q03f).
compute R216Q03S=concat(R216Q03a to R216Q03c).
compute R219Q01S=concat(R219Q01a to R219Q01d).
compute R227Q02S=concat(R227Q02a to R227Q02g).

compute S128Q03S=concat(S128Q03a to S128Q03b).
compute S133Q04S=concat(S133Q04a to S133Q04c).
compute S209Q01S=concat(S209Q01a to S209Q01d).
compute S213Q01S=concat(S213Q01a to S213Q01d).
compute S252Q03S=concat(S252Q03a to S252Q03b).
compute S269Q04S=concat(S269Q04a to S269Q04d).
compute S270Q03S=concat(S270Q03a to S270Q03b).

recode M155Q04S ("    "="nnnn").
recode M192Q01S ("   "="nnn").
recode M266Q01S ("    "="nnnn").
recode R070Q07S ("     "="nnnnn").
recode R088Q04S ("     "="nnnnn").
recode R088Q05S ("   "="nnn").
recode R099Q03S ("   "="nnn").
recode R119Q09S ("  "="nn").
recode R120Q07S ("   "="nnn").
recode R122Q01S ("   "="nnn").
recode R122Q03S ("      "="nnnnnn").
recode R216Q03S ("   "="nnn").
recode R219Q01S ("    "="nnnn").
recode R227Q02S ("       "="nnnnnnn").
```

```
recode S128Q03S ("  "="nn").
recode S133Q04S ("   "="nnn").
recode S209Q01S ("    "="nnnn").
recode S213Q01S ("    "="nnnn").
recode S252Q03S ("  "="nn").
recode S269Q04S ("    "="nnn").
recode S270Q03S ("  "="nn").

missing values M155Q04S ("nnnn").
missing values M192Q01S ("nnn").
missing values M266Q01S ("nnnn").
missing values R070Q07S ("nnnnn").
missing values R088Q04S ("nnnnn").
missing values R088Q05S ("nnn").
missing values R099Q03S ("nnn").
missing values R119Q09S ("nn").
missing values R120Q07S ("nnn").
missing values R122Q01S ("nnn").
missing values R122Q03S ("nnnnnn").
missing values R216Q03S ("nnn").
missing values R219Q01S ("nnnn").
missing values R227Q02S ("nnnnnnn").
missing values S128Q03S ("nn").
missing values S133Q04S ("nnn").
missing values S209Q01S ("nnnn").
missing values S213Q01S ("nnnn").
missing values S252Q03S ("nn").
missing values S269Q04S ("nnnn").
missing values S270Q03S ("nn").


* Create single digit variables from the constructed strings. In general
  these are the number of matches, for instance the new variable is 3 if
  the student had 3 parts of a set of yes/no or multiple choice questions
  correct. Other questions aimed to capture information about different
  types of open ended response, and codes were assigned to these. For
  missing values, strings including only 8's or 9's were assigned the
  value '8' for the new variable, unless they were all nines in which case
  the value 9 was assigned. Blanks only arise from the merging of the
  data sets where students did not do booklets by design.

* Assigns number of correct yes/no parts in I043Q04 to I043K04 to create
  variable with values 0,1,2,3 with missing values 8 and 9 and a blank to
  indicate that the student was not assigned this question (did not sit a
  booklet in which it appeared) #ACCUM adds up number of matches and
  #MISSNG the number of missings. Note that in the substr function #S1 is
  a substring of length 1 (the third argument) starting at position #I of
  the value of the variable given as the first argument.


compute #works='2111'.
compute #NACHK =INDEX(M155Q04S,'n').
Do If (#NACHK = 0).
compute #ACCUM=0.
compute #MISSNG=0.
Loop #I=1 to 4.
+ compute #S1=substr(M155Q04S,#I,1).
+ compute #S2=substr(#works,#I,1).
If (#S1 EQ #S2) #ACCUM=#ACCUM+1.
If (#S1 EQ '8' or #S1 EQ '9') #MISSNG=#MISSNG+1.
End loop.
compute M155Q04T=string(#ACCUM,F1.0).
if (#MISSNG eq 4) M155Q04T='8'.
if (M155Q04S eq '9999') M155Q04T='9'.
if (M155Q04S eq '    ') M155Q04T=' '.
Else If (#NACHK > 0).
compute M155Q04T='n'.
End If.

compute #works='162'.
```

```
compute #NACHK =INDEX(M192Q01S,'n').
Do If (#NACHK = 0).
compute #ACCUM=0.
compute #MISSNG=0.
Loop #I=1 to 3.
+ compute #S1=substr(M192Q01S,#I,1).
+ compute #S2=substr(#works,#I,1).
If (#S1 EQ #S2) #ACCUM=#ACCUM+1.
If (#S1 EQ '8' or #S1 EQ '9') #MISSNG=#MISSNG+1.
End loop.
compute M192Q01T=string(#ACCUM,F1.0).
if (#MISSNG eq 3) M192Q01T='8'.
if (M192Q01S eq '999') M192Q01T='9'.
if (M192Q01S eq '   ') M192Q01T=' '.
Else If (#NACHK > 0).
compute M192Q01T='n'.
End If.


compute #works='1211'.
compute #NACHK =INDEX(M266Q01S,'n').
Do If (#NACHK = 0).
compute #ACCUM=0.
compute #MISSNG=0.
Loop #I=1 to 4.
+ compute #S1=substr(M266Q01S,#I,1).
+ compute #S2=substr(#works,#I,1).
If (#S1 EQ #S2) #ACCUM=#ACCUM+1.
If (#S1 EQ '8' or #S1 EQ '9') #MISSNG=#MISSNG+1.
End loop.
compute M266Q01T=string(#ACCUM,F1.0).
if (#MISSNG eq 4) M266Q01T='8'.
if (M266Q01S eq '9999') M266Q01T='9'.
if (M266Q01S eq '    ') M266Q01T=' '.
Else If (#NACHK > 0).
compute M266Q01T='n'.
End If.


* Assigns number of correct yes/no parts in R070Q07 to R070K07 to create
  variable with values 0 t0 7 with missing values 8 and 9 and a blank to
  indicate that the student was not assigned this question (did not sit a
  booklet in which it appeared) #ACCUM adds up number of matches and
  #MISSNG the number of missings.


compute #works='21221'.
compute #NACHK =INDEX(R070Q07S,'n').
Do If (#NACHK = 0).
compute #ACCUM=0.
compute #MISSNG=0.
Loop #I=1 to 5.
+ compute #S1=substr(R070Q07S,#I,1).
+ compute #S2=substr(#works,#I,1).
If (#S1 EQ #S2) #ACCUM=#ACCUM+1.
If (#S1 EQ '8' or #S1 EQ '9') #MISSNG=#MISSNG+1.
End loop.
compute R070Q07T=string(#ACCUM,F1.0).
if (#MISSNG eq 5) R070Q07T='8'.
if (R070Q07S eq '99999') R070Q07T='9'.
if (R070Q07S eq '     ') R070Q07T=' '.
Else If (#NACHK > 0).
compute R070Q07T='n'.
End If.


compute #works='13234'.
compute #NACHK =INDEX(R088Q04S,'n').
Do If (#NACHK = 0).
compute #ACCUM=0.
```

```
compute #MISSNG=0.
Loop #I=1 to 5.
+ compute #S1=substr(R088Q04S,#I,1).
+ compute #S2=substr(#works,#I,1).
If (#S1 EQ #S2) #ACCUM=#ACCUM+1.
If (#S1 EQ '8' or #S1 EQ '9') #MISSNG=#MISSNG+1.
End loop.
compute R088Q04T=string(#ACCUM,F1.0).
if (#MISSNG eq 5) R088Q04T='8'.
if (R088Q04S eq '99999') R088Q04T='9'.
if (R088Q04S eq '     ') R088Q04T=' '.
Else If (#NACHK > 0).
compute R088Q04T='n'.
End If.



compute #works='112'.
compute #NACHK =INDEX(R088Q05S,'n').
Do If (#NACHK = 0).
compute #ACCUM=0.
compute #MISSNG=0.
          Loop #I=1 to 3.
+ compute #S1=substr(R088Q05S,#I,1).
+ compute #S2=substr(#works,#I,1).
If (#S1 EQ #S2) #ACCUM=#ACCUM+1.
If (#S1 EQ '8' or #S1 EQ '9') #MISSNG=#MISSNG+1.
End loop.
compute R088Q05T=string(#ACCUM,F1.0).
if (#MISSNG eq 3) R088Q05T='8'.
if (R088Q05S eq '999') R088Q05T='9'.
if (R088Q05S eq '   ') R088Q05T=' '.
Else If (#NACHK > 0).
compute R088Q05T='n'.
End If.


compute #works='113'.
compute #NACHK =INDEX(R099Q03S,'n').
Do If (#NACHK = 0).
compute #ACCUM=0.
compute #MISSNG=0.
Loop #I=1 to 3.
+ compute #S1=substr(R099Q03S,#I,1).
+ compute #S2=substr(#works,#I,1).
If (#S1 EQ #S2) #ACCUM=#ACCUM+1.
If (#S1 EQ '8' or #S1 EQ '9') #MISSNG=#MISSNG+1.
End loop.
compute R099Q03T=string(#ACCUM,F1.0).
if (#MISSNG eq 3) R099Q03T='8'.
if (R099Q03S eq '999') R099Q03T='9'.
if (R099Q03S eq '   ') R099Q03T=' '.
Else If (#NACHK > 0).
compute R099Q03T='n'.
End If.

compute #works='11'.
compute #NACHK =INDEX(R119Q09S,'n').
Do If (#NACHK = 0).
compute #ACCUM=0.
compute #MISSNG=0.
Loop #I=1 to 2.
+ compute #S1=substr(R119Q09S,#I,1).
+ compute #S2=substr(#works,#I,1).
If (#S1 EQ #S2) #ACCUM=#ACCUM+1.
If (#S1 EQ '8' or #S1 EQ '9') #MISSNG=#MISSNG+1.
End loop.
compute R119Q09T=string(#ACCUM,F1.0).
if (#MISSNG eq 2) R119Q09T='8'.
```

```
if (R119Q09S eq '99') R119Q09T='9'.
if (R119Q09S eq '  ') R119Q09T=' '.
Else If (#NACHK > 0).
compute R119Q09T='n'.
End If.


compute #works='211'.
compute #NACHK =INDEX(R120Q07S,'n').
Do If (#NACHK = 0).
compute #ACCUM=0.
compute #MISSNG=0.
Loop #I=1 to 3.
+ compute #S1=substr(R120Q07S,#I,1).
+ compute #S2=substr(#works,#I,1).
If (#S1 EQ #S2) #ACCUM=#ACCUM+1.
If (#S1 EQ '8' or #S1 EQ '9') #MISSNG=#MISSNG+1.
End loop.
compute R120Q07T=string(#ACCUM,F1.0).
if (#MISSNG eq 3) R120Q07T='8'.
if (R120Q07S eq '999') R120Q07T='9'.
if (R120Q07S eq '   ') R120Q07T=' '.
Else If (#NACHK > 0).
compute R120Q07T='n'.
End If.

compute #works='111'.
compute #NACHK =INDEX(R122Q01S,'n').
Do If (#NACHK = 0).
compute #ACCUM=0.
compute #MISSNG=0.
Loop #I=1 to 3.
+ compute #S1=substr(R122Q01S,#I,1).
+ compute #S2=substr(#works,#I,1).
If (#S1 EQ #S2) #ACCUM=#ACCUM+1.
If (#S1 EQ '8' or #S1 EQ '9') #MISSNG=#MISSNG+1.
End loop.
compute R122Q01T=string(#ACCUM,F1.0).
if (#ACCUM eq 3) R122Q01T='1'.
if (#ACCUM eq 1 or #ACCUM eq 2) R122Q01T='0'.
if (#MISSNG eq 3) R122Q01T='8'.
if (R122Q01S eq '999') R122Q01T='9'.
if (R122Q01S eq '   ') R122Q01T=' '.
Else If (#NACHK > 0).
compute R122Q01T='n'.
End If.

compute #works='212211'.
compute #NACHK =INDEX(R122Q03S,'n').
Do If (#NACHK = 0).
compute #ACCUM=0.
compute #MISSNG=0.
Loop #I=1 to 6.
+ compute #S1=substr(R122Q03S,#I,1).
+ compute #S2=substr(#works,#I,1).
If (#S1 EQ #S2) #ACCUM=#ACCUM+1.
If (#S1 EQ '8' or #S1 EQ '9') #MISSNG=#MISSNG+1.
End loop.
compute R122Q03T=string(#ACCUM,F1.0).
if (#MISSNG eq 6) R122Q03T='8'.
if (R122Q03S eq '999999') R122Q03T='9'.
if (R122Q03S eq '      ') R122Q03T=' '.
Else If (#NACHK > 0).
compute R122Q03T='n'.
End If.

compute #works='111'.
compute #NACHK =INDEX(R216Q03S,'n').
Do If (#NACHK = 0).
compute #ACCUM=0.
```

```
compute #MISSNG=0.
Loop #I=1 to 3.
+ compute #S1=substr(R216Q03S,#I,1).
+ compute #S2=substr(#works,#I,1).
If (#S1 EQ #S2) #ACCUM=#ACCUM+1.
If (#S1 EQ '8' or #S1 EQ '9') #MISSNG=#MISSNG+1.
End loop.
compute R216Q03T=string(#ACCUM,F1.0).
if (#ACCUM eq 3) R216Q03T='1'.
if (#ACCUM eq 1 or #ACCUM eq 2) R216Q03T='0'.
if (#MISSNG eq 3) R216Q03T='8'.
if (R216Q03S eq '999') R216Q03T='9'.
if (R216Q03S eq '   ') R216Q03T=' '.
Else If (#NACHK > 0).
compute R216Q03T='n'.
End If.


compute #works='1111'.
compute #NACHK =INDEX(R219Q01S,'n').
Do If (#NACHK = 0).
compute #ACCUM=0.
compute #MISSNG=0.
Loop #I=1 to 4.
+ compute #S1=substr(R219Q01S,#I,1).
+ compute #S2=substr(#works,#I,1).
If (#S1 EQ #S2) #ACCUM=#ACCUM+1.
If (#S1 EQ '8' or #S1 EQ '9') #MISSNG=#MISSNG+1.
End loop.
compute R219Q01T=string(#ACCUM,F1.0).
if (#ACCUM eq 4) R219Q01T='1'.
if (#ACCUM eq 1 or #ACCUM eq 2 or #ACCUM eq 3) R219Q01T='0'.
if (#MISSNG eq 4) R219Q01T='8'.
if (R219Q01S eq '9999') R219Q01T='9'.
if (R219Q01S eq '    ') R219Q01T=' '.
Else If (#NACHK > 0).
compute R219Q01T='n'.
End If.


compute #works='2121121'.
compute #NACHK =INDEX(R227Q02S,'n').
Do If (#NACHK = 0).
compute #ACCUM=0.
compute #MISSNG=0.
Loop #I=1 to 7.
+ compute #S1=substr(R227Q02S,#I,1).
+ compute #S2=substr(#works,#I,1).
If (#S1 EQ #S2) #ACCUM=#ACCUM+1.
If (#S1 EQ '8' or #S1 EQ '9') #MISSNG=#MISSNG+1.
End loop.
compute R227Q02T=string(#ACCUM,F1.0).
if (#MISSNG eq 7) R227Q02T='8'.
if (R227Q02S eq '9999999') R227Q02T='9'.
if (R227Q02S eq '       ') R227Q02T=' '.
Else If (#NACHK > 0).
compute R227Q02T='n'.
End If.


compute #works='12'.
compute #NACHK =INDEX(S128Q03S,'n').
Do If (#NACHK = 0).
compute #ACCUM=0.
compute #MISSNG=0.
Loop #I=1 to 2.
+ compute #S1=substr(S128Q03S,#I,1).
+ compute #S2=substr(#works,#I,1).
If (#S1 EQ #S2) #ACCUM=#ACCUM+1.
If (#S1 EQ '8' or #S1 EQ '9') #MISSNG=#MISSNG+1.
```

```
End loop.
compute S128Q03T=string(#ACCUM,F1.0).
if (#MISSNG eq 2) S128Q03T='8'.
if (S128Q03S eq '99') S128Q03T='9'.
if (S128Q03S eq '  ') S128Q03T=' '.
Else If (#NACHK > 0).
compute S128Q03T='n'.
End If.


compute #works='111'.
compute #NACHK =INDEX(S133Q04S,'n').
Do If (#NACHK = 0).
compute #ACCUM=0.
compute #MISSNG=0.
Loop #I=1 to 3.
+ compute #S1=substr(S133Q04S,#I,1).
+ compute #S2=substr(#works,#I,1).
If (#S1 EQ #S2) #ACCUM=#ACCUM+1.
If (#S1 EQ '8' or #S1 EQ '9') #MISSNG=#MISSNG+1.
End loop.
compute S133Q04T=string(#ACCUM,F1.0).
if (#MISSNG eq 3) S133Q04T='8'.
if (S133Q04S eq '999') S133Q04T='9'.
if (S133Q04S eq '   ') S133Q04T=' '.
Else If (#NACHK > 0).
compute S133Q04T='n'.
End If.


compute #works='1121'.
compute #NACHK =INDEX(S209Q01S,'n').
Do If (#NACHK = 0).
compute #ACCUM=0.
compute #MISSNG=0.
Loop #I=1 to 4.
+ compute #S1=substr(S209Q01S,#I,1).
+ compute #S2=substr(#works,#I,1).
If (#S1 EQ #S2) #ACCUM=#ACCUM+1.
If (#S1 EQ '8' or #S1 EQ '9') #MISSNG=#MISSNG+1.
End loop.
compute S209Q01T=string(#ACCUM,F1.0).
if (#MISSNG eq 4) S209Q01T='8'.
if (S209Q01S eq '9999') S209Q01T='9'.
if (S209Q01S eq '    ') S209Q01T=' '.
Else If (#NACHK > 0).
compute S209Q01T='n'.
End If.

compute #works='1112'.
compute #NACHK =INDEX(S213Q01S,'n').
Do If (#NACHK = 0).
compute #ACCUM=0.
compute #MISSNG=0.
Loop #I=1 to 4.
+ compute #S1=substr(S213Q01S,#I,1).
+ compute #S2=substr(#works,#I,1).
If (#S1 EQ #S2) #ACCUM=#ACCUM+1.
If (#S1 EQ '8' or #S1 EQ '9') #MISSNG=#MISSNG+1.
End loop.
compute S213Q01T=string(#ACCUM,F1.0).
if (#MISSNG eq 4) S213Q01T='8'.
if (S213Q01S eq '9999') S213Q01T='9'.
if (S213Q01S eq '    ') S213Q01T=' '.
Else If (#NACHK > 0).
compute S213Q01T='n'.
End If.

compute #works='21'.
compute #NACHK =INDEX(S252Q03S,'n').
```

```
Do If (#NACHK = 0).
compute #ACCUM=0.
compute #MISSNG=0.
Loop #I=1 to 2.
+ compute #S1=substr(S252Q03S,#I,1).
+ compute #S2=substr(#works,#I,1).
If (#S1 EQ #S2) #ACCUM=#ACCUM+1.
If (#S1 EQ '8' or #S1 EQ '9') #MISSNG=#MISSNG+1.
End loop.
compute S252Q03T=string(#ACCUM,F1.0).
if (#MISSNG eq 2) S252Q03T='8'.
if (S252Q03S eq '99') S252Q03T='9'.
if (S252Q03S eq '  ') S252Q03T=' '.
Else If (#NACHK > 0).
compute S252Q03T='n'.
End If.

compute #works='2112'.
compute #NACHK =INDEX(S269Q04S,'n').
Do If (#NACHK = 0).
compute #ACCUM=0.
compute #MISSNG=0.
Loop #I=1 to 4.
+ compute #S1=substr(S269Q04S,#I,1).
+ compute #S2=substr(#works,#I,1).
If (#S1 EQ #S2) #ACCUM=#ACCUM+1.
If (#S1 EQ '8' or #S1 EQ '9') #MISSNG=#MISSNG+1.
End loop.
compute S269Q04T=string(#ACCUM,F1.0).
if (#MISSNG eq 4) S269Q04T='8'.
if (S269Q04S eq '9999') S269Q04T='9'.
if (S269Q04S eq '    ') S269Q04T=' '.
Else If (#NACHK > 0).
compute S269Q04T='n'.
End If.

compute #works='21'.
compute #NACHK =INDEX(S270Q03S,'n').
Do If (#NACHK = 0).
compute #ACCUM=0.
compute #MISSNG=0.
Loop #I=1 to 2.
+ compute #S1=substr(S270Q03S,#I,1).
+ compute #S2=substr(#works,#I,1).
If (#S1 EQ #S2) #ACCUM=#ACCUM+1.
If (#S1 EQ '8' or #S1 EQ '9') #MISSNG=#MISSNG+1.
End loop.
compute S270Q03T=string(#ACCUM,F1.0).
if (#MISSNG eq 2) S270Q03T='8'.
if (S270Q03S eq '99') S270Q03T='9'.
if (S270Q03S eq '  ') S270Q03T=' '.
Else If (#NACHK > 0).
compute S270Q03T='n'.
End If.

* Double digit to score (not intermediate categories because of country
*  differences).


do if (M124Q03 = '97').
compute M124Q03T = 'n'.
else if (M124Q03 = '99').
compute M124Q03T = '9'.
else if (substr(M124Q03,1,1)='0').
compute M124Q03T = '0'.
else if (substr(M124Q03,1,1)='1').
compute M124Q03T = '1'.
else if (substr(M124Q03,1,1)='2').
compute M124Q03T = '2'.
else if (substr(M124Q03,1,1)='3').
```

```
compute M124Q03T = '3'.
end if.

do if (M136Q01 = '97').
compute M136Q01T = 'n'.
else if (M136Q01 = '99').
compute M136Q01T = '9'.
else if (substr(M136Q01,1,1)='0').
compute M136Q01T = '0'.
else if (substr(M136Q01,1,1)='1').
compute M136Q01T = '1'.
else if (substr(M136Q01,1,1)='2').
compute M136Q01T = '2'.
end if.

do if (M136Q02 = '97').
compute M136Q02T = 'n'.
else if (M136Q02 = '99').
compute M136Q02T = '9'.
else if (substr(M136Q02,1,1)='0').
compute M136Q02T = '0'.
else if (substr(M136Q02,1,1)='1').
compute M136Q02T = '1'.
end if.

do if (M136Q03 = '97').
compute M136Q03T = 'n'.
else if (M136Q03 = '99').
compute M136Q03T = '9'.
else if (substr(M136Q03,1,1)='0').
compute M136Q03T = '0'.
else if (substr(M136Q03,1,1)='1').
compute M136Q03T = '1'.
else if (substr(M136Q03,1,1)='2').
compute M136Q03T = '2'.
end if.

do if (M148Q02 = '97').
compute M148Q02T = 'n'.
else if (M148Q02 = '99').
compute M148Q02T = '9'.
else if (substr(M148Q02,1,1)='0').
compute M148Q02T = '0'.
else if (substr(M148Q02,1,1)='1').
compute M148Q02T = '1'.
else if (substr(M148Q02,1,1)='2').
compute M148Q02T = '2'.
end if.

do if (M150Q02 = '97').
compute M150Q02T = 'n'.
else if (M150Q02 = '99').
compute M150Q02T = '9'.
else if (substr(M150Q02,1,1)='0').
compute M150Q02T = '0'.
else if (substr(M150Q02,1,1)='1').
compute M150Q02T = '1'.
else if (substr(M150Q02,1,1)='2').
compute M150Q02T = '2'.
end if.

do if (M150Q03 = '97').
compute M150Q03T = 'n'.
else if (M150Q03 = '99').
compute M150Q03T = '9'.
else if (substr(M150Q03,1,1)='0').
compute M150Q03T = '0'.
else if (substr(M150Q03,1,1)='1').
compute M150Q03T = '1'.
end if.
```

```
do if (M155Q02 = '97').
compute M155Q02T = 'n'.
else if (M155Q02 = '99').
compute M155Q02T = '9'.
else if (substr(M155Q02,1,1)='0').
compute M155Q02T = '0'.
else if (substr(M155Q02,1,1)='1').
compute M155Q02T = '1'.
else if (substr(M155Q02,1,1)='2').
compute M155Q02T = '2'.
end if.

do if (M155Q03 = '97').
compute M155Q03T = 'n'.
else if (M155Q03 = '99').
compute M155Q03T = '9'.
else if (substr(M155Q03,1,1)='0').
compute M155Q03T = '0'.
else if (substr(M155Q03,1,1)='1').
compute M155Q03T = '1'.
else if (substr(M155Q03,1,1)='2').
compute M155Q03T = '2'.
end if.

do if (M179Q01 = '97').
compute M179Q01T = 'n'.
else if (M179Q01 = '99').
compute M179Q01T = '9'.
else if (substr(M179Q01,1,1)='0').
compute M179Q01T = '0'.
else if (substr(M179Q01,1,1)='1').
compute M179Q01T = '1'.
else if (substr(M179Q01,1,1)='2').
compute M179Q01T = '2'.
end if.

do if (S114Q03 = '97').
compute S114Q03T = 'n'.
else if (S114Q03 = '99').
compute S114Q03T = '9'.
else if (substr(S114Q03,1,1)='0').
compute S114Q03T = '0'.
else if (substr(S114Q03,1,1)='1').
compute S114Q03T = '1'.
end if.

do if (S114Q04 = '97').
compute S114Q04T = 'n'.
else if (S114Q04 = '99').
compute S114Q04T = '9'.
else if (substr(S114Q04,1,1)='0').
compute S114Q04T = '0'.
else if (substr(S114Q04,1,1)='1').
compute S114Q04T = '1'.
else if (substr(S114Q04,1,1)='2').
compute S114Q04T = '2'.
end if.

do if (S114Q05 = '97').
compute S114Q05T = 'n'.
else if (S114Q05 = '99').
compute S114Q05T = '9'.
else if (substr(S114Q05,1,1)='0').
compute S114Q05T = '0'.
else if (substr(S114Q05,1,1)='1').
compute S114Q05T = '1'.
end if.

do if (S129Q02 = '97').
```

```
compute S129Q02T = 'n'.
else if (S129Q02 = '99').
compute S129Q02T = '9'.
else if (substr(S129Q02,1,1)='0').
compute S129Q02T = '0'.
else if (substr(S129Q02,1,1)='1').
compute S129Q02T = '1'.
else if (substr(S129Q02,1,1)='2').
compute S129Q02T = '2'.
end if.

do if (S131Q02 = '97').
compute S131Q02T = 'n'.
else if (S131Q02 = '99').
compute S131Q02T = '9'.
else if (substr(S131Q02,1,1)='0').
compute S131Q02T = '0'.
else if (substr(S131Q02,1,1)='1').
compute S131Q02T = '1'.
end if.

do if (S131Q04 = '97').
compute S131Q04T = 'n'.
else if (S131Q04 = '99').
compute S131Q04T = '9'.
else if (substr(S131Q04,1,1)='0').
compute S131Q04T = '0'.
else if (substr(S131Q04,1,1)='1').
compute S131Q04T = '1'.
end if.

do if (S195Q02 = '97').
compute S195Q02T = 'n'.
else if (S195Q02 = '99').
compute S195Q02T = '9'.
else if (substr(S195Q02,1,1)='0').
compute S195Q02T = '0'.
else if (substr(S195Q02,1,1)='1').
compute S195Q02T = '1'.
else if (substr(S195Q02,1,1)='2').
compute S195Q02T = '2'.
end if.

do if (S195Q05 = '97').
compute S195Q05T = 'n'.
else if (S195Q05 = '99').
compute S195Q05T = '9'.
else if (substr(S195Q05,1,1)='0').
compute S195Q05T = '0'.
else if (substr(S195Q05,1,1)='1').
compute S195Q05T = '1'.
end if.

do if (S209Q02 = '97').
compute S209Q02T = 'n'.
else if (S209Q02 = '99').
compute S209Q02T = '9'.
else if (substr(S209Q02,1,1)='0').
compute S209Q02T = '0'.
else if (substr(S209Q02,1,1)='1').
compute S209Q02T = '1'.
end if.

do if (S253Q01 = '97').
compute S253Q01T = 'n'.
else if (S253Q01 = '99').
compute S253Q01T = '9'.
else if (substr(S253Q01,1,1)='0').
compute S253Q01T = '0'.
else if (substr(S253Q01,1,1)='1').
```

```
compute S253Q01T = '1'.
else if (substr(S253Q01,1,1)='2').
compute S253Q01T = '2'.
else if (substr(S253Q01,1,1)='3').
compute S253Q01T = '3'.
end if.

do if (S268Q02 = '97').
compute S268Q02T = 'n'.
else if (S268Q02 = '99').
compute S268Q02T = '9'.
else if (substr(S268Q02,1,1)='0').
compute S268Q02T = '0'.
else if (substr(S268Q02,1,1)='1').
compute S268Q02T = '1'.
end if.

do if (S269Q03 = '97').
compute S269Q03T = 'n'.
else if (S269Q03 = '99').
compute S269Q03T = '9'.
else if (substr(S269Q03,1,1)='0').
compute S269Q03T = '0'.
else if (substr(S269Q03,1,1)='1').
compute S269Q03T = '1'.
end if.


* Two string variables converted to 0,1,9,n variables.


do if (M273Q01 = '9997').
compute M273Q01T = 'n'.
else if (M273Q01 = '9999').
compute M273Q01T = '9'.
else if (M273Q01='4213').
compute M273Q01T = '1'.
else.
compute M273Q01T = '0'.
end if.

do if (M145Q01 = '999997').
compute M145Q01T = 'n'.
else if (M145Q01 = '999999').
compute M145Q01T = '9'.
else if (M145Q01='154265').
compute M145Q01T = '1'.
else.
compute M145Q01T = '0'.
end if.

* Here numerical values are coded as 1 if correct, 0 if incorrect and
* 9 if missing, n if Non applicable. For example the missing value for
* M034Q01 is 99, converted to 9 for the single character M034Q01T.

MISSING VALUES M034Q01 M037Q01 M037Q02 M144Q01 M144Q02 M144Q04 ().

compute M034Q01T='n'.
do if (M034Q01=21).
compute M034Q01T='1'.
else if (M034Q01=99).
compute M034Q01T='9'.
else if (M034Q01=97).
compute M034Q01T='n'.
else if (not sysmis(M034Q01)).
compute M034Q01T='0'.
end if.

compute M144Q01T='n'.
do if (M144Q01=27).
```

```
compute M144Q01T='1'.
else if (M144Q01=99).
compute M144Q01T='9'.
else if (M144Q01=97).
compute M144Q01T='n'.
else if (not sysmis(M144Q01)).
compute M144Q01T='0'.
end if.

compute M144Q02T='n'.
do if (M144Q02=12).
compute M144Q02T='1'.
else if (M144Q02=99).
compute M144Q02T='9'.
else if (M144Q02=97).
compute M144Q02T='n'.
else if (not sysmis(M144Q02)).
compute M144Q02T='0'.
end if.

compute M144Q04T='n'.
do if (M144Q04=8).
compute M144Q04T='1'.
else if (M144Q04=99).
compute M144Q04T='9'.
else if (M144Q04=97).
compute M144Q04T='n'.
else if (not sysmis(M144Q04)).
compute M144Q04T='0'.
end if.

compute M037Q01T='n'.
do if (M037Q01=144).

compute M037Q01T='1'.
else if (M037Q01=999).
compute M037Q01T='9'.
else if (M037Q01=997).
compute M037Q01T='n'.
else if (not sysmis(M037Q01)).
compute M037Q01T='0'.
end if.

compute M037Q02T='n'.
do if (M037Q02=6).
compute M037Q02T='1'.
else if (M037Q02=99).
compute M037Q02T='9'.
else if (M037Q02=97).
compute M037Q02T='n'.
else if (not sysmis(M037Q02)).
compute M037Q02T='0'.
end if.


execute.

missing values M034Q01     ( ,97,99).
missing values M037Q01     ( ,997,999).
missing values M037Q02     ( ,97,99).
missing values M144Q01     ( ,97,99).
missing values M144Q02     ( ,97,99).
missing values M144Q04     ( ,97,99).
missing values M034Q01T    ("n","8","9").
missing values M037Q01T    ("n","8","9").
missing values M037Q02T    ("n","8","9").
missing values M124Q03T    ("n","8","9").
missing values M136Q01T    ("n","8","9").
missing values M136Q02T    ("n","8","9").
missing values M136Q03T    ("n","8","9").
```

```
missing values M144Q01T    ("n","8","9").
missing values M144Q02T    ("n","8","9").
missing values M144Q04T    ("n","8","9").
missing values M145Q01T    ("n","8","9").
missing values M148Q02T    ("n","8","9").
missing values M150Q02T    ("n","8","9").
missing values M150Q03T    ("n","8","9").
missing values M155Q02T    ("n","8","9").
missing values M155Q03T    ("n","8","9").
missing values M155Q04T    ("n","8","9").
missing values M179Q01T    ("n","8","9").
missing values M192Q01T    ("n","8","9").
missing values M266Q01T    ("n","8","9").
missing values M273Q01T    ("n","8","9").
missing values R070Q07T    ("n","8","9").
missing values R088Q04T    ("n","8","9").
missing values R088Q05T    ("n","8","9").
missing values R099Q03T    ("n","8","9").
missing values R119Q09T    ("n","8","9").
missing values R120Q07T    ("n","8","9").
missing values R122Q01T    ("n","8","9").
missing values R122Q03T    ("n","8","9").
missing values R216Q03T    ("n","8","9").
missing values R219Q01T    ("n","8","9").
missing values R227Q02T    ("n","8","9").
missing values S114Q03T    ("n","8","9").
missing values S114Q04T    ("n","8","9").
missing values S114Q05T    ("n","8","9").
missing values S128Q03T    ("n","8","9").
missing values S129Q02T    ("n","8","9").
missing values S131Q02T    ("n","8","9").
missing values S131Q04T    ("n","8","9").
missing values S133Q04T    ("n","8","9").
missing values S195Q02T    ("n","8","9").
missing values S195Q05T    ("n","8","9").
missing values S209Q01T    ("n","8","9").
missing values S209Q02T    ("n","8","9").
missing values S213Q01T    ("n","8","9").
missing values S252Q03T    ("n","8","9").
missing values S253Q01T    ("n","8","9").
missing values S268Q02T    ("n","8","9").
missing values S269Q03T    ("n","8","9").
missing values S269Q04T    ("n","8","9").
missing values S270Q03T    ("n","8","9").
```

Appendix 5: Scores allocated to the items

		S=1	S=2	S=3
1	M033Q01	4		
2	M034Q01T	1		
3	M037Q01T	1		
4	M037Q02T	1		
5	M124Q01	2		
6	M124Q03T	1	2	3
7	M136Q01T	2		
8	M136Q02T	1		
9	M136Q03T	1	2	
10	M144Q01T	1		
11	M144Q02T	1		
12	M144Q03	1		
13	M144Q04T	1		
14	M145Q01T	1		
15	M148Q02T	1	2	
16	M150Q01	1		
17	M150Q02T	1	2	
18	M150Q03T	1		
19	M155Q01	1		
20	M155Q02T	1	2	
21	M155Q03T	1	2	
22	M155Q04T	4		
23	M159Q01	2		
24	M159Q02	3		
25	M159Q03	2		
26	M159Q05	2		
27	M161Q01	4		
28	M179Q01T	1	2	
29	M192Q01T	2,3		
30	M266Q01T	4		
31	M273Q01T	1		
32	R040Q02	1		
33	R040Q03A	1		
34	R040Q03B	1		
35	R040Q04	1		
36	R040Q06	3		
37	R055Q01	4		
38	R055Q02	1		
39	R055Q03	2		
40	R055Q05	1		
41	R061Q01	1,2		
42	R061Q03	3		
43	R061Q04	3		
44	R061Q05	1		
45	R067Q01	3		
46	R067Q04	1	2	
47	R067Q05	1	2	
48	R070Q02	1		
49	R070Q03	3		
50	R070Q04	1		
51	R070Q07T	4	5	
52	R076Q03	1		
53	R076Q04	1		
54	R076Q05	1		
55	R077Q02	2		
56	R077Q03	1	2	
57	R077Q04	2		
58	R077Q05	2		
59	R077Q06	4		
60	R081Q01	2		
61	R081Q05	1		
62	R081Q06A	1		
63	R081Q06B	1		
64	R083Q01	4		
65	R083Q02	1		

		S=1	S=2	S=3
66	R083Q03	1		
67	R083Q04	1		
68	R083Q06	1		
69	R086Q04	1		
70	R086Q05	3		
71	R086Q07	1		
72	R088Q01	4		
73	R088Q03	1	2	
74	R088Q04T	3,4	5	
75	R088Q05T	3		
76	R088Q07	3		
77	R091Q05	1		
78	R091Q06	2		
79	R091Q07B	2		
80	R093Q03	1		
81	R099Q04B	2	3	
82	R100Q04	2		
83	R100Q05	3		
84	R100Q06	3		
85	R100Q07	2		
86	R101Q01	3		
87	R101Q02	2		
88	R101Q03	2		
89	R101Q04	3		
90	R101Q05	4		
91	R101Q08	3		
92	R102Q01	2		
93	R102Q04A	1		
94	R102Q05	1		
95	R102Q06	1		
96	R102Q07	3		
97	R104Q01	1		
98	R104Q02	1		
99	R104Q05	1	2	
100	R104Q06	1		
101	R110Q01	4		
102	R110Q04	1		
103	R110Q05	1		
104	R110Q06	4		
105	R111Q01	4		
106	R111Q02B	1	2	
107	R111Q04	3		
108	R111Q06B	1	2	
109	R119Q01	3		
110	R119Q04	3		
111	R119Q05	1	2,3	
112	R119Q06	2		
113	R119Q07	1,2	3	
114	R119Q08	1,2		
115	R119Q09T	1	2	
116	R120Q01	2		
117	R120Q03	1		
118	R120Q06	1		
119	R120Q07T	3		
120	R122Q02	4		
121	R122Q03T	5	6	
122	R216Q01	3		
123	R216Q02	1		
124	R216Q03T	1		
125	R216Q04	1		
126	R216Q06	4		
127	R219Q01T	1		
128	R219Q01E	1		
129	R219Q02	1		
130	R220Q01	1		

		S=1	S=2	S=3
131	R220Q02B	1		
132	R220Q04	4		
133	R220Q05	3		
134	R220Q06	3		
135	R225Q02	1		
136	R225Q03	2		
137	R225Q04	2		
138	R227Q01	2		
139	R227Q02T	5,6	7	
140	R227Q03	1		
141	R227Q04	1	2	
142	R227Q06	1		
143	R228Q01	4		
144	R228Q02	2		
145	R228Q04	4		
146	R234Q01	1		
147	R234Q02	1		
148	R236Q01	1		
149	R236Q02	1,2		
150	R237Q01	1		
151	R237Q03	1		
152	R238Q01	1		
153	R238Q02	1		
154	R239Q01	1		
155	R239Q02	1		
156	R241Q02	1		
157	R245Q01	1		
158	R245Q02	1		
159	R246Q01	1		
160	R246Q02	1		
161	S114Q03T	1		
162	S114Q04T	1	2	
163	S114Q05T	1		
164	S128Q01	1		
165	S128Q02	1		
166	S128Q03T	2		
167	S129Q01	1		
168	S129Q02T	1	2	
169	S131Q02T	1		
170	S131Q04T	1		
171	S133Q01	3		
172	S133Q03	1		
173	S133Q04T	3		
174	S195Q02T	1	2	
175	S195Q04	1		
176	S195Q05T	1		
177	S195Q06	2		
178	S209Q02T	1		
179	S213Q01T	4		
180	S213Q02	1		
181	S252Q01	3		
182	S252Q02	1		
183	S252Q03T	2		
184	S253Q01T	1	2,3	
185	S253Q02	2		
186	S253Q05	1		
187	S256Q01	1		
188	S268Q01	3		
189	S268Q02T	1		
190	S268Q06	2		
191	S269Q01	1		
192	S269Q03T	1		
193	S269Q04T	4		
194	S270Q03T	2		