HISTORICAL LIFE COURSE STUDIES

VOLUME 4
2017

EHPS
NETWORK

MISSION STATEMENT
HISTORICAL LIFE COURSE STUDIES

Historical Life Course Studies is the electronic journal of the *European Historical Population Samples Network* (EHPS-Net). The journal is the primary publishing outlet for research involved in the conversion of existing European and non-European large historical demographic databases into a common format, the Intermediate Data Structure, and for studies based on these databases. The journal publishes both methodological and substantive research articles.

Methodological Articles

This section includes methodological articles that describe all forms of data handling involving large historical databases, including extensive descriptions of new or existing databases, syntax, algorithms and extraction programs. Authors are encouraged to share their syntaxes, applications and other forms of software presented in their article, if pertinent, on the EHPS-Net website.

Research articles

This section includes substantive articles reporting the results of comparative longitudinal studies that are demographic and historical in nature, and that are based on micro-data from large historical databases.

Historical Life Course Studies is a no-fee double-blind, peer-reviewed open-access journal supported by the European Science Foundation (ESF, http://www.esf.org), the Scientific Research Network of Historical Demography (FWO Flanders, http://www.historicaldemography.be) and the International Institute of Social History Amsterdam (IISH, http://socialhistory.org/). Manuscripts are reviewed by the editors, members of the editorial and scientific boards, and by external reviewers. All journal content is freely available on the internet at http://www.ehps-net.eu/journal.

Editors: Koen Matthijs & Paul Puschmann
Family and Population Studies
KU Leuven, Belgium
hislives@kuleuven.be

The European Science Foundation (ESF) provides a platform for its Member Organisations to advance science and explore new directions for research at the European level. Established in 1974 as an independent non-governmental organisation, the ESF currently serves 78 Member Organisations across 30 countries. EHPS-Net is an ESF Research Networking Programme.

EUROPEAN
SCIENCE
FOUNDATION

The European Historical Population Samples Network (EHPS-net) brings together scholars to create a common format for databases containing non-aggregated information on persons, families and households. The aim is to form an integrated and joint interface between many European and non-European databases to stimulate comparative research on the micro-level.
Visit: http://www.ehps-net.eu.

EHPS
NETWORK

HISTORICAL LIFE COURSE STUDIES
VOLUME 4 (2017), 59-96 published 09-05-2017

IDS Transposer: A Users Guide

Emily Klancher Merchant
University of California at Davis and Dartmouth College

George Alter
University of Michigan

With the assistance of

Jane Wang
Ford Motor Company

Ashok Bhargav
University of Michigan

ABSTRACT

The Intermediate Data Structure (IDS) provides a standard format for storing and sharing individual-level longitudinal life-course data (Alter and Mandemakers 2014; Alter, Mandemakers and Gutmann 2009). Once the data are in the IDS format, a standard set of programs can be used to extract data for analysis, facilitating the analysis of data across multiple databases. Currently, life-course databases store information in a variety of formats, and the process of translating data into IDS can be long and tedious. The IDS Transposer is a software tool that automates this process for source data in any format, allowing database administrators to specify how their datasets are to be represented in IDS. This article describes how the IDS Transposer works, first by going through an example step-by-step, and then by discussing each part of the process and potential options and exceptions in detail.

Keywords: Longitudinal life-course data, Event history data, Data management, Family reconstitution, Historical demography, Intermediate Data Structure, IDS Transposer.

e-ISSN:	2352-6343
PID article:	http://hdl.handle.net/10622/23526343-2017-0004?locatt=view:master
PID Design Document and UG examples:	http://hdl.handle.net/10622/23526343-2017-0005?locatt=view:master

The article can be downloaded from here.

1 INTRODUCTION

It is important to note at the outset that there is no single correct way to represent a given dataset in IDS.[1] The process of translating a dataset from its native format into the IDS standard involves numerous decisions about how the particular dataset will be represented in IDS. Such decisions include designating Contexts, choosing Types and Relationships, and assigning dates in the Timestamp. The IDS Transposer retains all of the flexibility inherent in the IDS standard; it allows the user to specify exactly how a given dataset should be represented in IDS. For that reason, the user of the IDS Transposer must decide in advance how the original dataset will be represented in IDS. The Transposer does not make those decisions; it simply provides users with a simple tool for implementing them.

Step-by-Step Example:

In the following step by step example, we will refer to several tables which are included in the Appendix. We will work with three datasets: a) the "original" dataset, b) the output IDS dataset, and c) a modified "input" dataset with modifications to the "original" dataset required for IDS. The dataset to be translated to IDS, the "original dataset" and its modified counterpart the "input dataset". may include any number of tables. The resulting IDS tables (**INDIVIDUAL, CONTEXT, INDIV_INDIV, INDIV_CONTEXT, CONTEXT_CONTEXT**) will be referred to collectively as the "IDS dataset." The IDS Transposer takes two types of files, all in .csv (comma-separated values) or tab-delimited format: the input data files, which are created from but not necessarily identical to the tables of the original dataset; and two mapping files, titled *ENTITY* and *RELATIONSHIP*, which the user prepares to indicate how each element of the input data files is to be represented in the resulting IDS dataset. Following the IDS standard, file names, dataset names, and table names are in all uppercase (e.g. **INDIVIDUAL**), and field (column) names are in title case (e.g. **Type**). The contents of a cell within a table is given in quotes in the text (e.g. "Jones" or "45"), but not in the tables. The contents of the Type variable are all uppercase (e.g. "LAST_NAME"); the contents of other variables are in title case (e.g. "Jones"). Fields (columns) in the two mapping files used by the IDS Transposer (*ENTITY* and *RELATIONSHIP*) are italicized (e.g. *EntityType)*. Table names and fields/columns in IDS tables are in bold (e.g. **Id_D**).

For this example, we will use a synthetic dataset titled FAMREC, which was created to resemble datasets produced through family reconstitution. FAMREC includes information about the families of fictional couples born between 1700 and 1799 in the present-day neighborhoods of Los Angeles. This information is arrayed in two tables: the PARENTS table contains parents' first and last names, dates and locations of birth, marriage, and death, and occupations; the CHILDREN table contains children's first names and dates and locations of birth. Couples in the PARENTS table are identified by a unique identification code (Parid); children in the CHILDREN table are linked to their parents by the couple's identification code (Parid). Table 1a shows the information for the first five couples listed in the PARENTS table; Table 1b shows the information for their children in the CHILDREN table. Variable names and explanations for this example are given in Table 2, but in actual use variables may be given any names. Place names found in FAMREC are collected in the PLACES table shown in Table 1c.

The first couple in this example is Franklin Edwards, born March 28, 1760 in Glassell Park, and Nell Kim, born January 16, 1757 in Boyle Heights. They married in Watts on March 27, 1786. At the time, Nell was an eye doctor and Franklin was a salesperson. They had one child, Mona, born May 22, 1788 in Watts. Both spouses died in Watts, Franklin on November 25, 1798 and Nell on September 21, 1810. The second couple, Tristram Hernandez and Marina Kennedy, had seven children, Lora, Cameron, Roman, Judith, Roger, Stacy, and Bertha.

[1] The IDS Transposer is currently available as a web service supported by the Inter-university Consortium for Political and Social Research (ICPSR) at the University of Michigan at http://www.icpsr.umich.edu/icpsrweb/ICPSR/idsTransposer/idsTransposer. Software code for the IDS Transposer is available on Github at https://github.com/ICPSR/IDS. Jane Wang and Ashok Bhargav are the authors of the IDS Transposer software.

2 DESIRED IDS OUTPUT

The IDS Transposer requires the user to decide in advance how the original dataset will be represented in IDS. Tables 3a-e show how we will represent FAMREC in IDS for the purpose of this example, including all data for the first family in FAMREC.

In the **INDIVIDUAL** table (Table 3a), places are given with both names (**Value**) and context identification codes (**Value_Id_C**). Since **Value_Id_C** points directly to a row in the **CONTEXT** table, we do not need to put the place name in the **Value column,** but the place name is included to make the data easier to read.

IDS recognizes four kinds of **Date_Type** in the Timestamp attached to every attribute. "Event" is used when the value of an attribute, such as marital status, is known to have changed on a specific date. We use "Declared" when the value of an attribute is recorded in a document created on a specific date, but we do not know when that value was attained. When the timing of an event is provided in a document composed at a later date, we set **Date_Type** to "Reported." For example, it is common for marriage certificates to "report" birth dates for brides and grooms. "Assigned" is used when the database manager has imputed a date to an attribute.

"Marriage" is an "Event," when the date comes from the marriage record because the CIVIL_STATUS of the bride and groom transition from "Unmarried" to "Married" on that date. We know the OCCUPATIONs of the bride and groom on the day of the marriage from the marriage certificate, so the **Date_Type** for each occupation is "Declared". BIRTH_DATE appears in both "Event" and "Reported" **Date_Types**. The birth dates of the bride and groom are "Reported" retrospectively in the marriage certificate, but the birth dates of children come from birth certificates, recorded as each "Event" occurred. All of the dates in this dataset are known exactly (i.e. day, month, and year are present), so Estimation is set to "Exact".

We included a row for parents' SEX, which is not given in the original dataset, but can be inferred, assuming that all husbands are male and all wives female.

FAMREC locates events in two **CONTEXTs** (Table 3b): Unions and Neighborhoods. "Union" is not part of the IDS Standard, but we use it to illustrate the operation of the IDS Transposer. In this example "Union" refers to the marital union of a husband and wife, and it allows us to locate the conjugal family geographically over time. We generate one "Declared" record in the CONTEXT table for the Neighborhood in which the Union was located at the time of the marriage, at the birth of every child, and at the death of each spouse. Note that we are assuming that neither spouse remarries. If remarriage was possible, we would include only the earliest date of death of a spouse. We do not associate the locations and dates of children's marriages and deaths with the Union, because they may have left their parents' homes before those events occurred.

Contexts in IDS may be hierarchical, so we identify them with the attribute **Type** "LEVEL" (Table 3b). Unions have only one attribute **Type** ("LEVEL") while places also have "NAME". To illustrate nested contexts, we show the context hierarchy "Neighborhood", "Municipality", and "State". We have designated our places as "Time_Invariant", but start and end dates may be given in datasets when place names or boundaries change during the period observed in the data. Another alternative is to assign a known date to a context with **Date_Type** set to "Declared", which indicates that it was valid on that date but start and end dates are not known.

The **INDIV_INDIV** table (Table 3c) specifies all relationships between family members, in both directions (e.g., 10001 is the Husband of 10002 and 10002 is the wife of 10001). The start date of this relationship is the date of the marriage. We could have included the date of the first spouse to die as the end date of the relationship, but this date often needs to be computed. We listed child/parent relationships as "Time_Invariant", because parent-child relationships do not change once children are born.

The **INDIV_CONTEXT** table (Table 3d) places individuals in the lowest level context of the dataset, in this case the union (because unions are fully nested in neighborhoods). The Relation field specifies the relationship between the person and the context. We use marriage and birth dates as "declarations". However, a database administrator could assign start and end dates for the time that each person spends living in the family.

Finally, the **CONTEXT_CONTEXT** table (Table 3e) specifies the relationships among levels of context. We have sorted the rows to show the hierarchy of state-municipality-neighborhood. The "Municipality and State" relationship only needs to be entered once to be applied to all neighborhoods within the city.

Union 1000 (Franklin Edwards and Nell Kim) generates four rows in **CONTEXT_CONTEXT** (table 3e), all of which place the union in Watts (ID_C=104). We see records in 1786 when the couple married, 1788 when their child was born, 1798 when the husband died, and 1810 when the wife died. There are ten observations for union 1001 (Tristram Hernandez and Marina Kennedy). This couple was married and had four children in Tarzana (ID_C=128). Between 1786 and 1788 they moved to Arletta (ID_C=107) where their last three children were born. They were still in Arletta when the husband and wife died. We assign the **Date_Type** "Declared" to these rows, because we only know that these families were in those neighborhoods on specific days.

As this example demonstrates, there are many decisions to be made when translating a dataset into IDS. The *ENTITY* and *RELATIONSHIP* mapping files allow the user to specify to the IDS Transposer how the original dataset should be represented in IDS.

3 DATA PREPARATION

The desired output in our example IDS tables (Tables 3a-3e) includes information that is not in our original dataset. For example, Table 1a shows an identification number for the marriage (parid), but the husband and wife do not have IDs as individuals. For this example, we have kept the additional information to the minimum: we added unique identification codes for people and places, and we have added the sex of the parents. We use Parid, which is the identifier for a marital union, as the context identifier (**ID_C**) for the union. Tables 4a-4c show the input data files in which these additional fields have been added to the original data (PARENTS_INPUT, CHILDREN_INPUT, PLACES_INPUT).

To prepare the input data files, we must do the following:

- Create a table of place names and context identification codes.
- Add context identification codes to matching place names.
- Create unique identification codes for husbands, wives, and children.
- Attach husband and wife identification codes to their children.
- Split date variables into their components: day, month, year.
- Create a variable indicating the source of data for each row. If the source of the data is not given in the dataset, we specify the name of the file in the original dataset (PARENTS or CHILDREN).

The Appendix gives R code to create the table of place names, which we call PLACES_INPUT, and to add the requisite columns to the PARENTS and CHILDREN tables; we call the resulting tables PARENTS_INPUT and CHILDREN_INPUT. These tables are now our input data files for the IDS Transposer. If we had decided to include additional information, such as end of relationship dates or children's last names, we would also have added columns for that information to the input files. Note that we did not add columns for parents' sex: this information will be supplied in the mapping files.

4 MAPPING FILES

The IDS Transposer uses two "mapping" tables to control the transfer of information from the input data files to the output IDS tables. There are two mapping tables: *ENTITY* for the IDS **INDIVIDUAL** and **CONTEXT** tables and *RELATIONSHIP* for **INDIV_INDIV**, **INDIV_CONTEXT**, and **CONTEXT_CONTEXT** in IDS. Mapping is a complex process, because the structure of IDS (entity-attribute-value) is so different from the format in which most data are collected. As we illustrated above, a single row in the PARENTS file contributes many rows in the IDS **INDIVIDUAL** table.

In the following sections we offer a strategy for creating mapping files. It begins by manually converting a small sample of the original dataset into IDS. This sample IDS serves as a template for creating the mapping files. First, we identify the unique IDS data types found in the sample IDS and place them in the tabular format used by the mapping files. We then insert the fields in the input data files that will be required to populate the IDS tables. The steps in this mapping process are illustrated in the following four sets of tables:

A) Our sample IDS tables are shown in Tables 3a-e. These tables include a unique example of every IDS entry that can be created from the input tables. For example, we only need to convert one row in the CHILDREN file to IDS to show how all children will be mapped.

B) Tables (5 and 8) show how content is transposed between fields in the input tables and fields in the IDS output files. Table 5 illustrates the transpositions for the *ENTITY* tables (**INDIVIDUAL** and **CONTEXT**). Table 8 illustrates the transpositions for the *RELATIONSHIP* tables (**INDIV_INDIV, INDIV_CONTEXT**, and **CONTEXT_CONTEXT**).

C) Tables 6a-b and 9a-c serve as templates for the mapping files, *ENTITY* and *RELATIONSHIP*. These templates are constructed by copying the IDS control columns (e.g. **Type**, **Date_type**) from the sample IDS (Tables 3a-e) and ignoring the specific information from the input files that appears in our sample IDS. In Table 6a we list all of the IDS data types that will be derived for each type of person in the input data. The PARENTS file describes two "entities": husband and wife. Each row in the CHILDREN file describes a child. In Tables 9a-b we show *RELATIONSHIP* information derived from three sources: PARENTS, CHILDREN, and the PLACES file.

D) In Tables 7 and 10 we complete the mapping process by adding field names from the input files.

In the following two sections we will explain at greater length how the transposing process is completed, first for entities and then for the relationships.

4.1 *ENTITY* MAPPING FILE

Our example IDS tables in Table 3 provide a template for the mapping files, as shown in Table 6. We suppose here that Table 3 was created by hand from one row in each input data file (Table 1).

Tables 6a-b show how our example **INDIVIDUAL** table can be directly converted to rows in the *ENTITY* file: field names in the **INDIVIDUAL** and **CONTEXT** tables are given in boldface above the corresponding fields in the *ENTITY* file. Content of the cells in the *ENTITY* file that are identical to their values in the **INDIVIDUAL** or **CONTEXT** table are filled in; other cells are left blank. Note that the number of rows in *ENTITY* will be equivalent to the total number of rows for one individual of each role in the INDIVIDUAL table (10 rows for husband, 10 rows for wife, 4 rows for children in Table 6a) plus the number of rows in the **CONTEXT** table for one context of each level (1 row for union, 2 rows for place in Table 6b).

ENTITY does not include the IDS Id field, which is made automatically by the IDS Transposer.

ENTITY requires two fields that link input files to IDS tables:

- *tableName* tells the IDS Transposer the source of the data (PARENTS_INPUT, CHILDREN_INPUT, and PLACES_INPUT);
- *entityType* tells the IDS Transposer to which IDS table the output will be added (**INDIVIDUAL** or **CONTEXT**).

The Timestamp fields in the mapping file correspond directly to the Timestamp fields in the IDS tables:

- values for the IDS fields **Date_type**, **Estimation**, and **Missing** are given directly in the *DateType, DateEstimationType,* and *DateMissingType* fields of the mapping table;
- values of the other Timestamp fields come from the input data files.

Table 7 shows the completed *ENTITY* mapping file. Note that

- rows 1-10 indicate how the IDS Transposer is to create rows in the IDS **INDIVIDUAL** table for *each husband* in PARENTS_INPUT;
- rows 11-20 indicate how the IDS Transposer is to create rows in the IDS **INDIVIDUAL** table for *each wife* in PARENTS_INPUT;
- rows 21-24 indicate how the IDS Transposer is to create rows in the IDS **INDIVIDUAL** table for *each child* in CHILDREN_INPUT;
- row 25 indicates how the IDS Transposer is to create rows in the IDS **CONTEXT** table for *each union* in PARENTS_INPUT, and
- rows 26-27 indicate how the IDS Transposer is to create rows in the IDS **CONTEXT** table for *each place* in PLACES_INPUT.

Because rows 1-20 and 25 of the *ENTITY* mapping file come from the PARENTS_INPUT file, the field *TableName* takes the value "PARENTS_INPUT" for those rows. *TableName* takes the value "CHILDREN_INPUT" for rows 21-24, and "PLACES_INPUT" for rows 26-27. The field *EntityType* takes the value of "INDIVIDUAL" for rows 1-24, which specify how information should be displayed in the IDS **INDIVIDUAL** table, and "CONTEXT" for rows 25-27, which specify information destined for the **CONTEXT** table.

EntityID in the mapping file specifies the field in the input file giving an ID code for the relevant person. Since rows 1-10 create information about husbands, *EntityID* for those rows takes the value "hid," the name of the variable we created for the husband's identification code (Table 4a). Similarly, *EntityID* for rows 11-20 takes the value "wid" and *EntityID* for rows 21-24 takes the value "kidid." For the final three rows, the *EntityID* field takes the identification code for the relevant context: since row 25 refers to a union, its identification code is "parid" from PARENTS_INPUT; rows 26-27 refer to places, so *EntityID* takes the value of "placeid," the identification code we gave to places in PLACES_INPUT.

The **Source** field in an IDS table may be used to identify different source documents within a database. The IDS Transposer expects a *Source* column in the input data to provide this information. In this example, we created a variable called Source_table in all input data files. If the name of this variable varied across input files, *Source* could take different values, as in the example given below (see 5. An Alternative Data Structure).

The *type* field in *ENTITY* is identical to the **Type** field in the IDS tables. Note that types are listed separately for each role in the original dataset (e.g., FIRST_NAME is listed once for husbands, once for wives, and once for children).

Information that will go into the **Value** field in the IDS table is listed either in *VariableName* or *Value* or *Value_ID_C* in the mapping file. Either the VariableName field or the Value field will contain information; the others will be blank. For types that refer only to a date, such as BIRTH_DATE, all three fields will be blank because the relevant information is a date described by the 'time stamp'.

- If the information for the IDS **Value** field is in the input data file, the name of the relevant variable/column is specified in the *VariableName* field in the mapping file. For example, the value of FIRST_NAME for husbands is contained in the variable Hfirst, so we list that in the *Variable-*

Name column of the first row of the mapping file. Note that in row 11, which produces the FIRST_NAME row for all wives, we specify "wfirst," the variable containing wives' first names in PARENTS_INPUT, in the *VariableName* field of the mapping file.

- If the information for the IDS **Value** field is not in the input data file, but is the same for all individuals in a role (that is, all husbands, all wives, or all children), the value is specified directly in the *Value* field of the mapping table. In our example, the sex of parents is not given in PARENTS_INPUT. Assuming that all husbands are male and all wives female, we enter "Male" in the Value field of the mapping file for row 3 and "Female" in the *Value* field of the mapping file for row 13. The text given in the *Value* column is assigned to the person identified in the *entityID* column: "hid" for husbands and "wid" for wives.
- If the IDS value for a given **Type** is a context identification code (**Value_Id_C**), the field in the input data set containing that code is listed in the *Value_ID_C* field of the mapping file. In our example, husband's birth location is a context, given in the variable Hbirthloc_id in PARENTS_INPUT, so we list "Hbirthloc_id" in the *Value_ID_C* field.

For rows 25-27, which produce the **CONTEXT** table, the context identification code goes in the *EntityID* field of the mapping file, as described above, and the *Value_ID_C* field is blank. The value for LEVEL is specified directly in the *Value* field of the mapping file; the names of places are contained in the Place field in PLACES_INPUT, so "Place" is listed in the *VariableName* field of the mapping file.

For the Timestamp, the values of the IDS fields **Date_type** and **Missing** are specified directly in the corresponding fields of the entity mapping file (*DateType* and *DateMissingType*). Consequently, these fields will be the same for every IDS row created from that *ENTITY* mapping. Thus, all rows describing the husband's FIRST_NAME in IDS will get the **Date_type** and **Missing** provided in the first row of the *ENTITY* table.

Births, marriages, and deaths are listed as "Event" in the *DateType* field, because they represent transitions occurring on a specific day. Occupations are assigned to *DateType* "Declared," because we do not know when an individual changed occupation.

The value of the **Missing** field in the IDS tables is specified directly only if it is the same for the given combination of **Type** and role. For example, "Time_Invariant" is specified in row 3 for SEX, because we assume that all husbands are male at all times.

The *DateEstimationType* field is used to indicate that a date or period in the timestamp has been estimated by the database provider.

- When a field/column in the input file is specified in *DateEstimationType*, the IDS Transposer uses values from this field for all rows of this type. For example, setting *DateEstimationType* to "est_var" tells IDS Transposer to take values for the **Estimation** field from that column in the input file.
- If no field/column is listed in *DateEstimationType*, the IDS Transposer will create this variable from the date fields. IDS Transposer uses "Exact" if all components of the date are present in the data, "Year_Month" if the day is missing, and "Year" if only year is given. These are specific to individuals, so if one person has only the year of his birth listed in the input data file and another has the full date, **Estimation** in the resulting IDS **INDIVIDUAL** table will be "Year" for the first and "Exact" for the second.

The fields *Day, Month, Year, Startday, Startmonth, Startyear, Endday, Endmonth*, and *Endyear* in the mapping file take the names of the fields in the input data file containing the relevant day, month, and year. Date variables are to be listed for each relevant combination of Type and structural role. In our example, "Hbirth_day" is given in the *Day* field of the mapping file for the husband's BIRTH_DATE and the husband's BIRTH_LOCATION. Because we assume that information about occupation was collected from the marriage certificate, we list the marriage date fields in the OCCUPATION rows.

Note that we use the date of the marriage (Marriage_day, Marriage_month, Marriage_year) to create a record in the **CONTEXT** table for each marital "union." In this case we put the marriage date in IDS fields *Day, Month, Year*, which we identify as an "event." Marriage is the starting date for a marital union, but we did not use the starting date fields (*Startday, Startmonth, Startyear*), because we have not computed ending dates for each union. IDS expects a starting date to have a corresponding ending date. **Periods that are not completely defined may cause problems for programs designed to extract data from IDS.**[2]

4.2 RELATIONSHIP MAPPING FILE

As in the *ENTITY* file, each row in the *RELATIONSHIP* file creates a row in an IDS table for every row with relevant information in the input data files. Table 8 shows the relationship between fields in the *RELATIONSHIP* file and the fields that they fill in the **INDIV_INDIV, INDIV_CONTEXT**, and **CONTEXT_CONTEXT** files. We fill templates for the *RELATIONSHIP* table by selecting the minimum number of rows to fill the IDS tables in Tables 3c, 3d, and 3e. Templates for the *RELATIONSHIP* table are shown in Table 9.

Table 9a shows six rows destined for the **INDIV_INDIV** table. There are three dyadic pairs (husband-wife, father-child, mother-child), and each pair enters the table twice (e.g. "husband" and "wife"). Relationships between individuals are listed in a reciprocal way, once for the relationship of the first person to the second and again for the relationship of the second person to the first (e.g., spousal relationships will be listed twice, once with "Husband" as the *Relation* and once with "Wife" as the *Relation*). The marriages described in the PARENTS table yield one relationship pair, husband-wife, and the births described in the CHILDREN table provide two pairs, mother-child and father-child.

Table 9b shows how we place each type of person in the **INDIV_CONTEXT** table. The three rows in this table represent the roles in a marital union: husband, wife, child. Relationships between individuals and contexts are listed once, with the *Relation* field referring to the relationship of the individual to the context

In Table 9c we represent five rows with **CONTEXT_CONTEXT** relationships. Relationships between contexts are listed once, with the lower level given first and the Relation field referring to the relationship of the lower level context to the higher level context. The template has one row for the hierarchy of places (Neighborhood/Municipality/State) that is described in the PLACES_INPUT file (see below). **CONTEXT_CONTEXT** information placing the family (marital union) in a neighborhood is recorded at four key events: marriage, husband's death, wife's death, and the births of children. Each of these events generates a "Union and Neighborhood" row in **CONTEXT_CONTEXT**.

The templates in Table 9 are completed in the *RELATIONSHIP* table shown in Table 10. In the *RELATIONSHIP* table rows 1-2 refer to the spousal relationship, rows 3-6 refer to relationships between children and parents. Rows 7-8 place husbands and wives in marital unions, and row 9 associates children with the unions of their parents.

Row 10 is used to read the hierarchy contexts from the PLACE_INPUT file. Note that we added three columns to PLACES_INPUT that were not in the original PLACES file: Level, Nested_in, and Relvar. The Nested_in column is the ID of the next higher level in the context hierarchy, and Relvar is the relationship of this level to the next higher level. For example, "Los Angeles" is a "Municipality", which is nested in the "State" of "California". The relationship between "Los Angeles" and "California" is "Municipality and State". Since the relationship is read from PLACES_INPUT, we put Relvar in the *Relation_Variable* column of the *RELATIONSHIP* file.

Locations of "Union in Neighborhood" are derived from the PARENTS_INPUT and CHILDREN_INPUT files in Rows 11-14. Row 11 in the *RELATIONSHIP* table identifies the location of the union at the time of the marriage. Rows 12 and 13 show where the family was living when the husband and wife died. Row 14 will add a row to **CONTEXT_CONTEXT** each time that a child is born. We use *DateType* "Declared" for these rows, because we do not know if and when the location of the union changed.

[2] The ending date of a marital union is not a settled research question. Does it end at the first death of a spouse? Does it end when the wife can no longer bear children? Are children living with a widowed mother in the same household or a different one? The relevant ending date depends upon the type of question a researcher intends to answer.

Note that we are locating individuals in places by associating them with unions. We could also choose to associate individuals directly with places, by using their IDs in the **INDIV_CONTEXT** table, but the approach shown here is more efficient. Since individuals are linked to unions, each "Union in Neighborhood" row in the **CONTEXT_CONTEXT** table automatically associates every person in the family with the new neighborhood.

Again, the *TableName* field is the name of the input data file that contains the information: PARENTS_INPUT for rows 1-2, 7-8, and 10-13 and CHILDREN_INPUT for rows 3-6 and 14. The *RelationshipType* field indicates the IDS table to which the information will be written: **INDIV_INDIV** for the spousal and parent-child relationships in rows 1-6, **INDIV_CONTEXT** for rows 7-9, and **CONTEXT_CONTEXT** for rows 10-14. These rows comprise the full *RELATIONSHIP* file, so all relationships of a given type are created in the same way. The *DatabaseID*, *Source*, and Timestamp fields in the *RELATIONSHIP* file are exactly the same as those in the *ENTITY* file and will not be discussed here.

The *FromEntityID* field in the *RELATIONSHIP* file points the IDS Transposer to the variable in the input data file that contains the identification code that will be the value of the **Id_I_1** field in the **INDIV_INDIV** table, the **Id_I** field in the **INDIV_CONTEXT** table, or the **Id_C_1** field in the **CONTEXT_CONTEXT** table. The *ToEntityID* field points to the variable in the input data file that contains the identification code that will be the value of the **Id_I_2** field in the **INDIV_INDIV** table, the **Id_C** field in the **INDIV_CONTEXT** table, or the **Id_C_2** field in the **CONTEXT_CONTEXT** table. Note that the ID variables shown in *FromEntityID* and *ToEntityID* fields are reversed as we move from "Husband" to "Wife" in rows 1 and 2.

The *RELATIONSHIP* file includes two fields that provide information for the **Relation** field in the resulting IDS tables: *Relation* and *RelationVariable*. Just as with the *VariableName* and *Value* fields in the *ENTITY* file, only one of these two fields may have a value in any given row. In our example, all but one of the relationships are specified directly in the mapping file through the *Relation* field. But in row 10 we use a *RelationVariable* to read from the Relvar column in the PLACES_INPUT file, which contains "Municipality and State" or "Neighborhood and Municipality".

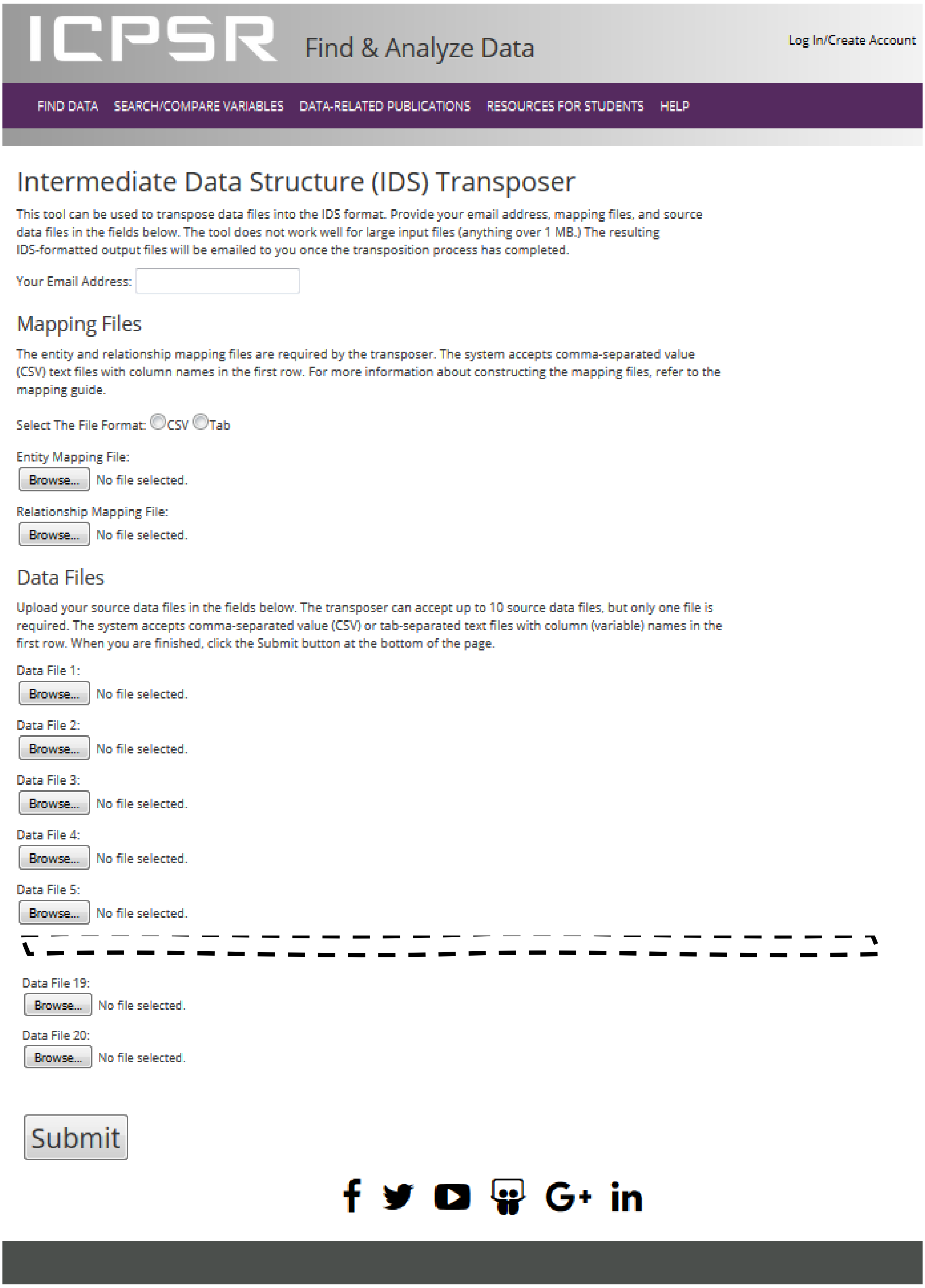
5 IDS TRANSPOSER

We now have all of the files necessary to use the IDS Transposer. Our input data files are the PARENTS_INPUT, CHILDREN_INPUT, and PLACE_INPUT tables described above in Step 2. Our mapping files are *ENTITY* and *RELATIONSHIP* shown in Table 7 and Table 10. Note that these tables include all of the rows necessary for mapping the FAMREL dataset in the format we created in Step 2. All files must be in .csv (comma-separated values) format. The IDS Transposer allows users to map to these files on their computer or server and emails the resulting IDS tables to users.

The IDS Transposer is found at this URL: http://www.icpsr.umich.edu/icpsrweb/ICPSR/idsTransposer/idsTransposer

Figure 1 shows a screen shot of the data entry page, and Figure 2 shows the result of a successful execution.

Figure 1 *IDS Transposer File Selection Screen*



Intermediate Data Structure (IDS) Transposer

This tool can be used to transpose data files into the IDS format. Provide your email address, mapping files, and source data files in the fields below. The tool does not work well for large input files (anything over 1 MB.) The resulting IDS-formatted output files will be emailed to you once the transposition process has completed.

Your Email Address:

Mapping Files

The entity and relationship mapping files are required by the transposer. The system accepts comma-separated value (CSV) text files with column names in the first row. For more information about constructing the mapping files, refer to the mapping guide.

Select The File Format: ○CSV ○Tab

Entity Mapping File:
Browse... No file selected.

Relationship Mapping File:
Browse... No file selected.

Data Files

Upload your source data files in the fields below. The transposer can accept up to 10 source data files, but only one file is required. The system accepts comma-separated value (CSV) or tab-separated text files with column (variable) names in the first row. When you are finished, click the Submit button at the bottom of the page.

Data File 1:
Browse... No file selected.

Data File 2:
Browse... No file selected.

Data File 3:
Browse... No file selected.

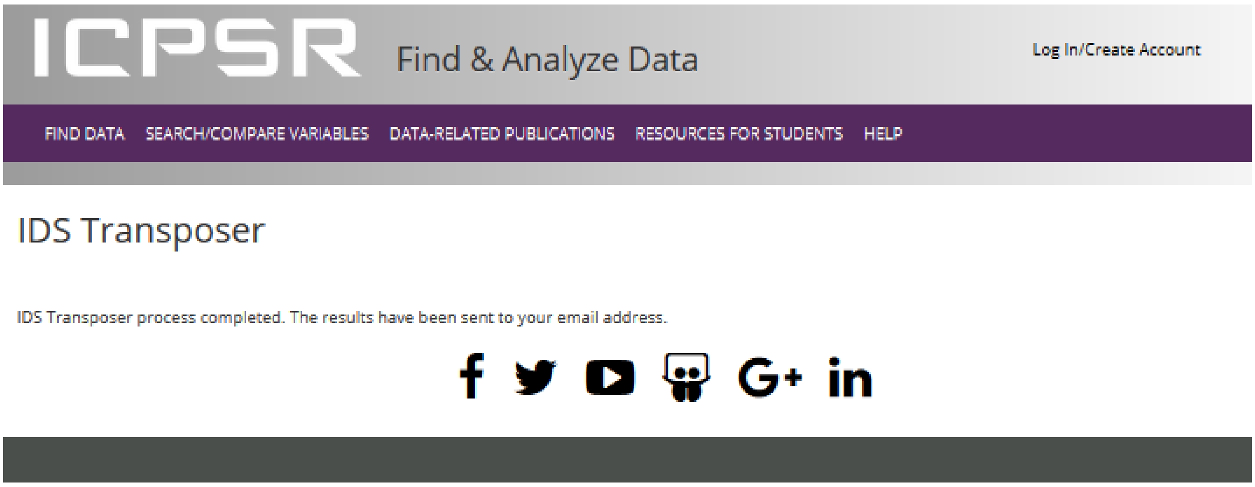
Data File 4:
Browse... No file selected.

Data File 5:
Browse... No file selected.

Data File 19:
Browse... No file selected.

Data File 20:
Browse... No file selected.

Figure 2 *IDS Transposer File Selection Screen*



IDS Transposer

IDS Transposer process completed. The results have been sent to your email address.

6 AN ALTERNATIVE DATA STRUCTURE

The example described above started with a file structure typically found in family reconstitutions studies. All information about a married couple is contained in one file, and their children are described in a second file. Here we show the same data in an alternative file structure where the data are arranged by source document: marriages, births, and deaths. Table 11 shows the input data files used in this example. Notice that the husband and wife in each marriage appear in all three files.

Table 12 shows the *ENTITY* mapping file for the data in Table 11. An important difference from the previous example is that names are given in all three input files. This is normally the case in the original sources, and it is common for names to differ between sources. For example, "William" in one source may be "Bill" in another. The IDS format described in Table 12 captures these variations, because names are taken from each input table. In IDS each version of a name is "Declared" with a date and a **Source** ("Births", "Marriages", or "Deaths").

The *RELATIONSHIP* mapping file in Table 13 differs from the previous example in its handling of contexts. In this example, we do not consider the marital union a context. Instead, we associate individuals with contexts on the date of each event through the **INDIV_CONTEXT** table.

7 CONCLUSION

The IDS Transposer is a powerful tool for moving data into the IDS standard. It efficiently changes the structure of a dataset in which one row represents an individual, a family, or an event to the entity-attribute-value system used in IDS, where one row represents the time-stamped value of a specific attribute. Typically, IDS data files are long and narrow with many more rows than the original data source. Thus, in our first example we began with three tables with a modest number of rows -- PARENTS (100 rows), CHILDREN (365 rows), PLACES (75 rows), and we ended with five much longer tables – **INDIVIDUAL** (3460 rows), **CONTEXT** (248 rows), **INDIV_INDIV** (1660 rows), **INDIV_CONTEXT** (565 rows), **CONTEXT_CONTEXT** (539 rows). The keys to using IDS Transposer are its two mapping files, *ENTITY* and *RELATIONSHIP*, which describe how the information in the original data files will be moved into IDS tables.

We have illustrated a methodology for creating these mapping files. First, we manually transform a sample from the original dataset into IDS. Then, we use this sample IDS to make templates for the *ENTITY* and *RELATIONSHIP* mapping files and to find the minimum number of rows required in each mapping file. Finally, we complete the mapping files by adding variable names and constant values. Note that we modified the original data files to create input files for IDS Transposer. In particular,

- we added IDs for persons and places that did not have separate IDs in the original files;
- date variables were separated into day, month, and year.

Clearly, preparation of the mapping files is much more time-consuming than running the IDS Transposer application, and we conclude here with some recommendations on creating those mappings.

- Use standard IDS **Types** and **Values** available from the IDS **METADATA** table whenever possible. Since IDS is intended to promote sharing of data management and data analysis software, following the standard will reduce the need to customize data extraction programs.
- Be aware that a single piece of information (especially a date) may be repeated on many rows in IDS.
- Be cautious in using periods (start date / end date) rather than single dates. Most of the sources used in historical demography do come from records associated with particular dates rather than periods of time. In particular, occupations and residences are usually known only at the time a document was created. If a man is recorded as a "carpenter" in his marriage certificate, we only know his occupation on that particular day. He may have been a "laborer" one year earlier and a "shopkeeper" one year later. Occupations are usually "declarations" that pertain to a specific date.
- Always check the IDS tables against the cases that you translated manually to be sure that you are getting the results that you expect. Run IDS validation and consistency tests to look for problems in the data or mapping files.

ACKNOWLEDGMENTS

This project was funded by a grant from the National Institute of Child Health and Human Development NIH grant number R21 HD060893 "Archiving the Historical Demography of the US". We wish to thank Kees Mandemakers for his helpful comments on this paper.

REFERENCES

Alter, G. & Mandemakers, K. (2014). The Intermediate Data Structure (IDS) for longitudinal historical microdata, version 4. *Historical Life Course Studies,* 1, 1-26.

Alter, G., Mandemakers, K. & Gutmann, M. (2009). Defining and distributing longitudinal historical data in a general way through an intermediate structure. *Historical Social Research,* 34(3), 78-114.

Appendix A Tables

Table 1a *Original data file PARENTS*

Parid	Hfirst	Hlast	Wfirst	Wlast	Hbirth	Wbirth	Marriage	Hdeath	Wdeath	Hocc	Wocc	Hbirthloc	Wbirthloc	Marrloc	Hdeathloc	Wdeathloc
1000	Franklin	Edwards	Nell	Kim	1760-03-28	1757-01-16	1786-03-27	1798-11-25	1810-09-21	Salesperson	Eye Doctor	Glassell Park	Boyle Heights	Watts	Watts	Watts
1001	Tristram	Hernandez	Marina	Kennedy	1752-08-02	1753-03-05	1778-05-24	1796-10-25	1817-05-23	Police	Banker	Chatsworth	Reynier	Tarzana	Arleta	Arleta
1002	Bradford	Fisher	Cynthia	Vasquez	1744-02-04	1743-11-28	1759-06-27	1804-10-02	1822-03-24	Historian	Film	Westwood	Larchmont	Glassell Park	Glassell Park	Glassell Park
1003	Oliver	Peterson	Antonia	Porter	1717-02-07	1714-08-14	1737-08-29	1806-09-08	1761-03-20	Designer	Maitre	Watts	Brookside	Montecito	Montecito	Montecito
1004	Morgan	Lee	Adeline	Woods	1773-10-15	1773-11-24	1807-11-14	1819-10-07	1840-10-06	English	Science	Edendale	Cahuenga	Hollywood	Hollywood	Hollywood

Back to page 60

Table 1b *Original data file CHILDREN*

Parid	Name	Birth	Birthloc
1000	Mona	1788-05-22	Watts
1001	Lora	1780-07-27	Tarzana
1001	Cameron	1782-09-05	Tarzana
1001	Roman	1784-04-10	Tarzana
1001	Judith	1786-02-25	Tarzana
1001	Roger	1788-11-25	Arleta
1001	Stacy	1790-11-02	Arleta
1001	Bertha	1792-02-27	Arleta
1002	Vincent	1761-06-05	Glassell Park
1002	Darrell	1763-02-21	Glassell Park
1002	Lindon	1765-08-19	Glassell Park
1002	Noel	1767-01-05	Glassell Park
1003	Shauna	1739-11-06	Montecito
1003	Thelma	1741-02-11	Montecito
1004	Aurora	1809-09-24	Hollywood
1004	Shannon	1811-06-28	Hollywood

Back to page 60

Table 1c *Original data file PLACES*

place	placeid
California	1
Los Angeles	100
Glassell Park	101
Chatsworth	102
Westwood	103
Watts	104
Edendale	105

Back to page 60

Table 2a *Variable Names in PARENTS Table*

Variable name	Description
parid	Identification number of couple
hfirst	Husband's first name
hlast	Husband's last name
wfirst	Wife's first name
wlast	Wife's last name
hbirth	Husband's date of birth
wbirth	Wife's date of birth
marriage	Marriage date
hdeath	Husband's date of death
wdeath	Wife's date of death
hocc	Husband's occupation
wocc	Wife's occupation
hbirthloc	Husband's place of birth
wbirthloc	Wife's place of birth
marrloc	Place of marriage
hdeathloc	Husband's place of death
wdeathloc	Wife's place of death

Back to page 60

Table 2b *Variable Names in CHILDREN Table*

Variable name	Description
parid	Identification number of child's parents
name	Child's name
birth	Child's date of birth
birthloc	Child's place of birth

Back to page 60

Table 3a *INDIVIDUAL Table*

Id_D	Id_I	Source	Type	Value	Value_Id_C	Date_Type	Estimation	Missing	Year	Month	Day	Start_Year	Start_Month	Start_Day	End_Year	End_Month	End_Day
FAMREC	10001	PARENTS	FIRST_NAME	Franklin		Declared	Exact		1786	3	27						
FAMREC	10001	PARENTS	LAST_NAME	Edwards		Declared	Exact		1786	3	27						
FAMREC	10001	PARENTS	SEX	Male				Time_Invariant									
FAMREC	10001	PARENTS	OCCUPATION	Salesperson		Declared	Exact		1786	3	27						
FAMREC	10001	PARENTS	BIRTH_DATE			Event	Exact		1760	3	28						
FAMREC	10001	PARENTS	BIRTH_LOCATION	Glassell	101	Event	Exact		1760	3	28						
FAMREC	10001	PARENTS	MARRIAGE_DATE			Event	Exact		1786	3	27						
FAMREC	10001	PARENTS	MARRIAGE_LOCATION	Watts	104	Event	Exact		1786	3	27						
FAMREC	10001	PARENTS	DEATH_DATE			Event	Exact		1798	11	25						
FAMREC	10001	PARENTS	DEATH_LOCATION	Watts	104	Event	Exact		1798	11	25						
FAMREC	10002	PARENTS	FIRST_NAME	Nell			Exact		1786	3	27						
FAMREC	10002	PARENTS	LAST_NAME	Kim			Exact		1786	3	27						
FAMREC	10002	PARENTS	SEX	Female				Time_Invariant									
FAMREC	10002	PARENTS	OCCUPATION	Eye Doctor		Declared	Exact		1786	3	27						
FAMREC	10002	PARENTS	BIRTH_DATE			Event	Exact		1757	1	16						
FAMREC	10002	PARENTS	BIRTH_LOCATION	Boyle	136	Event	Exact		1757	1	16						
FAMREC	10002	PARENTS	MARRIAGE_DATE			Event	Exact		1786	3	27						
FAMREC	10002	PARENTS	MARRIAGE_LOCATION	Watts	104	Event	Exact		1786	3	27						
FAMREC	10002	PARENTS	DEATH_DATE			Event	Exact		1810	9	21						
FAMREC	10002	PARENTS	DEATH_LOCATION	Watts	104	Event	Exact		1810	9	21						
FAMREC	100001	CHILDREN	FIRST_NAME	Mona		Declared	Exact		1788	5	22						
FAMREC	100001	CHILDREN	LAST_NAME	Edwards		Declared	Exact		1788	5	22						
FAMREC	100001	CHILDREN	BIRTH_DATE			Event	Exact		1788	5	22						
FAMREC	100001	CHILDREN	BIRTH_LOCATION	Watts	104	Event	Exact		1788	5	22						

Back to page 61, 63

Table 3b *CONTEXT Table*

Id_D	Id_C	Source	Type	Value	Date_Type	Estimation	Missing	Year	Month	Day	Start_Year	Start_Month	Start_Day	End_Year	End_Month	End_Day
FAMREC	1000	PARENTS	LEVEL	Union	Declared	Exact		1786	3	27						
FAMREC	1001	PARENTS	LEVEL	Union	Declared	Exact		1778	5	24						
FAMREC	1002	PARENTS	LEVEL	Union	Declared	Exact		1759	6	27						
FAMREC	1003	PARENTS	LEVEL	Union	Declared	Exact		1737	8	29						
FAMREC	1004	PARENTS	LEVEL	Union	Declared	Exact		1807	11	14						
FAMREC	1		LEVEL	State			Time_Invariant									
FAMREC	1		NAME	California			Time_Invariant									
FAMREC	100		LEVEL	Municipality			Time_Invariant									
FAMREC	100		NAME	Los Angeles			Time_Invariant									
FAMREC	101		LEVEL	Neighborhood			Time_Invariant									
FAMREC	101		NAME	Glassell Park			Time_Invariant									
FAMREC	104		LEVEL	Neighborhood			Time_Invariant									
FAMREC	104		NAME	Watts			Time_Invariant									
FAMREC	136		LEVEL	Neighborhood			Time_Invariant									
FAMREC	136		NAME	Boyle Heights			Time_Invariant									

Back to page 61, 63

Table 3c. *INDIV_INDIV Table*

Id_D	Id_I_1	Id_I_2	Source	Relation	Date_Type	Estimation	Missing	Year	Month	Day	Start_Year	Start_Month	Start_Day	End_Year	End_Month	End_Day
FAMREC	10001	10002	PARENTS	Husband	Declared	Exact		1786	3	27						
FAMREC	10002	10001	PARENTS	Wife	Declared	Exact		1786	3	27						
FAMREC	10001	100001	CHILDREN	Father		Exact	Time_Invariant									
FAMREC	100001	10001	CHILDREN	Child		Exact	Time_Invariant									
FAMREC	10002	100001	CHILDREN	Mother		Exact	Time_Invariant									
FAMREC	100001	10002	CHILDREN	Child		Exact	Time_Invariant									

Back to page 61, 63, 66

Table 3d *INDIV_CONTEXT*

Id_D	Id_I	Id_C	Source	Relation	Date_Type	Estimation	Missing	Year	Month	Day	Start_Year	Start_Month	Start_Day	End_Year	End_Month	End_Day
FAM-REC	10001	1000	PARENTS	Husband	Declared	Exact		1786	3	27						
FAM-REC	10002	1000	PARENTS	Wife	Declared	Exact		1786	3	27						
FAM-REC	100001	1000	CHILDREN	Child	Declared	Exact		1788	5	22						

Back to page 61, 63, 66

Table 3e *CONTEXT_CONTEXT*

Id_D	Id_C_1	Id_C_2	Source	Relation	Date_Type	Estimation	Missing	Year	Month	Day	Start_Year	Start_Month	Start_Day	End_Year	End_Month	End_Day
FAMREC	100	1		Municipality and State			Time_Invariant									
FAMREC	101	100		Neighborhood and Municipality			Time_Invariant									
FAMREC	102	100		Neighborhood and Municipality			Time_Invariant									
FAMREC	103	100		Neighborhood and Municipality			Time_Invariant									
FAMREC	104	100		Neighborhood and Municipality			Time_Invariant									
FAMREC	1000	104	PARENTS	Union and Neighborhood	Declared	Exact		1786	3	27						
FAMREC	1000	104	CHILDREN	Union and Neighborhood	Declared	Exact		1788	5	22						
FAMREC	1000	104	PARENTS	Union and Neighborhood	Declared	Exact		1798	11	25						
FAMREC	1000	104	PARENTS	Union and Neighborhood	Declared	Exact		1810	9	21						
FAMREC	1001	128	PARENTS	Union and Neighborhood	Declared	Exact		1778	5	24						
FAMREC	1001	128	CHILDREN	Union and Neighborhood	Declared	Exact		1780	7	27						

Table 3e continued on next page

Table 3e *CONTEXT_CONTEXT (continued)*

Id_D	Id_C_1	Id_C_2	Source	Relation	Date_Type	Estimation	Missing	Year	Month	Day	Start_Year	Start_Month	Start_Day	End_Year	End_Month	End_Day
FAMREC	1001	128	CHILDREN	Union and Neighborhood	Declared	Exact		1782	9	5						
FAMREC	1001	128	CHILDREN	Union and Neighborhood	Declared	Exact		1784	4	10						
FAMREC	1001	128	CHILDREN	Union and Neighborhood	Declared	Exact		1786	2	25						
FAMREC	1001	107	CHILDREN	Union and Neighborhood	Declared	Exact		1788	11	25						
FAMREC	1001	107	CHILDREN	Union and Neighborhood	Declared	Exact		1790	11	2						
FAMREC	1001	107	CHILDREN	Union and Neighborhood	Declared	Exact		1792	2	27						
FAMREC	1001	107	PARENTS	Union and Neighborhood	Declared	Exact		1796	10	25						
FAMREC	1001	107	PARENTS	Union and Neighborhood	Declared	Exact		1817	5	23						

Back to page 62, 63, 66

Table 4a *Variables in the PARENTS_INPUT table*

Variable name	Description	Source for computed variables
parid	Identification number of couple	
hfirst	Husband's first name	
hlast	Husband's last name	
wfirst	Wife's first name	
wlast	Wife's last name	
hbirth	Husband's date of birth	
wbirth	Wife's date of birth	
marriage	Marriage date	
hdeath	Husband's date of death	

Table 4a continued on next page

Table 4a *Variables in the PARENTS_INPUT table (continued)*

Variable name	Description	Source for computed variables
wdeath	Wife's date of death	
hocc	Husband's occupation	
wocc	Wife's occupation	
hbirthloc	Husband's place of birth	
wbirthloc	Wife's place of birth	
marrloc	Place of marriage	
hdeathloc	Husband's place of death	
wdeathloc	Wife's place of death	
hid	Identification number of husband	1000*parid+1
wid	Identification number of wife	1000*parid+2
source_table	Data source	Set to "PARENTS"
hbirth_day	Husband's date of birth: Day	Day from hbirth
hbirth_month	Husband's date of birth: Month	Month from hbirth
hbirth_year	Husband's date of birth: Year	Year from hbirth
wbirth_day	Wife's date of birth: Day	Day from wbirth
wbirth_month	Wife's date of birth: Month	Month from wbirth
wbirth_year	Wife's date of birth: Year	Year from wbirth
marriage_day	Date of marriage: Day	Day from marriage
marriage_month	Date of marriage: Month	Month from marriage
marriage_year	Date of marriage: Year	Year from marriage
hdeath_day	Husband's date of death: Day	Day from hdeath
hdeath_month	Husband's date of death: Month	Month from hdeath
hdeath_year	Husband's date of death: Year	Year from hdeath
wdeath_day	Wife's date of death: Day	Day from wdeath
wdeath_month	Wife's date of death: Month	Month from wdeath
wdeath_year	Wife's date of death: Year	Year from wdeath
hbirthloc_id	Identification number for husband's birth location	Id_C from CONTEXT table for hbirthloc
wbirthloc_id	Identification number for wife's birth location	Id_C from CONTEXT table for wbirthloc
marrloc_id	Identification number for marriage location	Id_C from CONTEXT table for marrloc
hdeathloc_id	Identification number for husband's death location	Id_C from CONTEXT table for hdeathloc
wdeathloc_id	Identification number for wife's death location	Id_C from CONTEXT table for wdeathloc

Back to page 62

Table 4b *Variables in the CHILDREN_INPUT table*

Variable name	Description	Source for computed variables
parid	Identification number of child's parents as a couple	
name	Child's name	
lastname	Last name of father	hlast in PARENTS table
birth	Child's date of birth	
birthloc	Child's place of birth	
kidid	Identification number of child	10000*parid+[birth order within family]
source_table	Data source	Set to "Children"
momid	Identification number for child's mother	1000*parid+2
dadid	Identification number for child's father	1000*parid+1
birth_day	Child's date of birth: Day	Day from birth
birth_month	Child's date of birth: Month	Month from birth
birth_year	Child's date of birth: Year	Year from birth
birthloc_id	Identification number for child's birth location	Id_C from CONTEXT table for birthloc

Back to page 62

Table 4c *Variables in the PLACES_INPUT table*

Variable name	Description	Source for computed variables
place	Name of place	Unique values from PARENTS and CHILDREN
placeid	Context identification code for place	Assigned numeric values
level	Level in context hierarchy	"Neighborhood," "Municipality," or "State"
nested_in	Next higher context level	placeid of context in which this place is nested
relvar	Relation to next higher context level ("Neighborhood in Municipality," "Municipality in state"	"Neighborhood in Municipality," or "Municipality in State"

Back to page 62

Table 5 *Fields in IDS INDIVIDUAL and CONTEXT Tables and IDS Transposer ENTITY Mapping File*

IDS INDIVIDUAL and CONTEXT Tables		**IDS Transposer *ENTITY* Mapping File**	
Field name	**Description**	**Field name**	**Use**
Id	Primary key		Assigned automatically by IDS Transposer
		TableName	Name of data file for input. "PARENTS" is interpreted as "PARENTS.csv"
		EntityType	INDIVIDUAL or CONTEXT
Id_D	Identifier of the database or parts of the database from which the data are extracted.	*DatabaseID*	Short text describing the database
Id_I	Identifying number of each individual in the database	*EntityID*	Name of a column in the input file with an ID assigned to this individual or context.
Source	Specification of the source.	*Source*	Short text describing the source of the data within the database.
Type	Type of attribute	*Type*	An IDS attribute type.
Value	The value of the attribute.	*VariableName*	Name of a column in the input file with a value for each row
		Value	Short text to be assigned to every row
Value_Id_C	Identifier to the CONTEXT-table for values of a contextual nature.	*Value_ID_C*	Name of a column in the input file with the ID of a context.
Date_type	Event, Reported, Declared, Assigned	*DateType*	An IDS date_type: Event, Reported, Declared, Assigned
Estimation	Type of estimate of the date or period (e.g. "Age_based," "Middling")	*DateEstimationType*	When this field is blank, IDS Transposer will compute a value using the date information provided ("Exact," "Month_year", "Year"). The user may also specify a column in the input file giving the estimation method for each row.
Day	Day number	*Day*	Name of a column in the input file
Month	Month number	*Month*	Name of a column in the input file
Year	Year number	*Year*	Name of a column in the input file
Start_day	Start day number	*StartDay*	Name of a column in the input file
Start_month	Start month number	*StartMonth*	Name of a column in the input file
Start_year	Start year number	*StartYear*	Name of a column in the input file
End_day	End day number	*EndDay*	Name of a column in the input file
End_month	End month number	*EndMonth*	Name of a column in the input file
End_year	End year number	*EndYear*	Name of a column in the input file
Missing	This field explains why a date or part of a date is missing	*DateMissingType*	Short text explaining why the date is missing, e.g. "Unavailable," "Time_Invariant"

Back to page 63

Table 6a *INDIVIDUAL Table as Template for ENTITY File*

		Id_D	**Id_I**	**Source**	**Type**	**Value**		**Value_ID_C**	**Date_type**	**Estimation**	**Day**	**Month**	**Year**	**Start_day**	**Start_month**	**Start_year**	**End_day**	**End_month**	**End_year**	**Missing**
TableName	*EntityType*	*DatabaseID*	*EntityID*	*Source*	*Type*	*VariableName*	*Value*	*Value_ID_C*	*DateType*	*DateEstimation Type*	*Day*	*Month*	*Year*	*StartDay*	*StartMonth*	*StartYear*	*EndDay*	*EndMonth*	*EndYear*	*DateMissingType*
		FAMREC			FIRST_NAME				Declared											
		FAMREC			LAST_NAME				Declared											
		FAMREC			SEX															Time_Invariant
		FAMREC			OCCUPATION				Event											
		FAMREC			BIRTH_DATE				Declared											
		FAMREC			BIRTH_LOCATION				Event											
		FAMREC	Husband		MARRIAGE_DATE				Event											
		FAMREC			MARRIAGE_LOCATION				Event											
		FAMREC			DEATH_DATE				Event											
		FAMREC			DEATH_LOCATION															
		FAMREC			FIRST_NAME				Declared											
		FAMREC			LAST_NAME				Declared											
		FAMREC			SEX															Time_Invariant
		FAMREC			OCCUPATION				Event											
		FAMREC			BIRTH_DATE				Declared											
		FAMREC	Wife		BIRTH_LOCATION				Event											
		FAMREC			MARRIAGE_DATE				Event											

Table 6a *INDIVIDUAL Table as Template for ENTITY File (continued)*

		Id_D	**Id_I**	**Source**	**Type**	**Value**		**Value_ID_C**	**Date_type**	**Estimation**	**Day**	**Month**	**Year**	**Start_day**	**Start_month**	**Start_year**	**End_day**	**End_month**	**End_year**	**Missing**
TableName	*Entity-Type*	*DatabaseID*	*EntityID*	*Source*	*Type*	*VariableName*	*Value*	*Value_ID_C*	*DateType*	*DateEstimation Type*	*Day*	*Month*	*Year*	*StartDay*	*StartMonth*	*StartYear*	*EndDay*	*EndMonth*	*EndYear*	*DateMissingType*
		FAMREC	Wife		MARRIAGE_LOCATION				Event											
		FAMREC			DEATH_DATE				Event											
		FAMREC			DEATH_LOCATION															
		FAMREC	Child		FIRST_NAME				Declared											
		FAMREC			LAST_NAME				Declared											
		FAMREC			BIRTH_DATE				Event											
		FAMREC			BIRTH_LOCATION				Event											

Back to page 63

Table 6b *CONTEXT Table as Template for Rows in ENTITY File*

		Id_D	**Id_C**	**Source**	**Type**	**Value**			**Date_type**	**Estimation**	**Day**	**Month**	**Year**	**Start_day**	**Start_month**	**Start_year**	**End_day**	**End_month**	**End_year**	**Missing**
TableName	*Entity-Type*	*DatabaseID*	*Entity-ID*			*VariableName*	*Value*	*Value_ID_C*	*DateType*	*DateEstimation-Type*	*Day*	*Month*	*Year*	*StartDay*	*StartMonth*	*StartYear*	*EndDay*	*EndMonth*	*EndYear*	*DateMissingType*
		FAMREC	Union		LEVEL															
		FAMREC	Place		LEVEL				Declared											Time_Invariant
		FAMREC			NAME				Declared											Time_Invariant

Back to page 63

Table 7 *Mapping File: ENTITY*

TableName	*Entity Type*	Database ID	*EntityID*	Source	Type	Variable Name	Value	Value_ ID_C	DateType	DateEstimationType	Day	Month	Year	Start Day	Start Month	Start Year	EndDay	End Month	End Year	Date MissingType
PARENTS_ INPUT	INDIVIDUAL	FAMREC	hid	source_ table	FIRST_ NAME	hfirst			Declared		marriage_ day	marriage_ month	marriage_ year							
PARENTS_ INPUT	INDIVIDUAL	FAMREC	hid	source_ table	LAST_ NAME	hlast			Declared		marriage_ day	marriage_ month	marriage_ year							
PARENTS_ INPUT	INDIVIDUAL	FAMREC	hid	source_ table	SEX		Male													Time_ Invariant
PARENTS_ INPUT	INDIVIDUAL	FAMREC	hid	source_ table	OCCUPATION	hocc			Declared		marriage_ day	marriage_ month	marriage_ year							
PARENTS_ INPUT	INDIVIDUAL	FAMREC	hid	source_ table	BIRTH_ DATE				Event		hbirth_ day	hbirth_ month	hbirth_ year							
PARENTS_ INPUT	INDIVIDUAL	FAMREC	hid	source_ table	BIRTH_ LOCATION	hbirthloc		hbirthloc_id	Event		hbirth_ day	hbirth_ month	hbirth_ year							
PARENTS_ INPUT	INDIVIDUAL	FAMREC	hid	source_ table	MARRIAGE_ DATE				Event		marriage_ day	marriage_ month	marriage_ year							
PARENTS_ INPUT	INDIVIDUAL	FAMREC	hid	source_ table	MARRIAGE_ LOCATION	marrloc		marrloc_id	Event		marriage_ day	marriage_ month	marriage_ year							
PARENTS_ INPUT	INDIVIDUAL	FAMREC	hid	source_ table	DEATH_ DATE				Event		hdeath_ day	hdeath_ month	hdeath_ year							
PARENTS_ INPUT	INDIVIDUAL	FAMREC	hid	source_ table	DEATH_ LOCATION	hdeathloc		hdeathloc_id	Event		hdeath_ day	hdeath_ month	hdeath_ year							
PARENTS_ INPUT	INDIVIDUAL	FAMREC	wid	source_ table	FIRST_ NAME	wfirst			Declared		marriage_ day	marriage_ month	marriage_ year							
PARENTS_ INPUT	INDIVIDUAL	FAMREC	wid	source_ table	LAST_ NAME	wlast			Declared		marriage_ day	marriage_ month	marriage_ year							
PARENTS_ INPUT	INDIVIDUAL	FAMREC	wid	source_ table	SEX		Female													Time_ Invariant
PARENTS_ INPUT	INDIVIDUAL	FAMREC	wid	source_ table	OCCUPATION	wocc			Declared		marriage_ day	marriage_ month	marriage_ year							

Table 7 *Mapping File: ENTITY (continued)*

TableName	*Entity Type*	Database ID	*Entity ID*	Source	Type	Variable Name	Value	Value_ID_C	DateType	DateEstimationType	Day	Month	Year	Start Day	Start Month	Start Year	EndDay	End Month	End Year	Date MissingType
PARENTS_INPUT	INDIVIDUAL	FAMREC	wid	source_table	BIRTH_DATE				Event		wbirth_day	wbirth_month	wbirth_year							
PARENTS_INPUT	INDIVIDUAL	FAMREC	wid	source_table	BIRTH_LOCATION	wbirthloc		wbirthloc_id	Event		wbirth_day	wbirth_month	wbirth_year							
PARENTS_INPUT	INDIVIDUAL	FAMREC	wid	source_table	MARRIAGE_DATE				Event		marriage_day	marriage_month	marriage_year							
PARENTS_INPUT	INDIVIDUAL	FAMREC	wid	source_table	MARRIAGE_LOCATION	Marrloc		marrloc_id	Event		marriage_day	marriage_month	marriage_year							
PARENTS_INPUT	INDIVIDUAL	FAMREC	wid	source_table	DEATH_DATE				Event		wdeath_day	wdeath_month	wdeath_year							
PARENTS_INPUT	INDIVIDUAL	FAMREC	wid	source_table	DEATH_LOCATION	wdeathloc		wdeathloc_id	Event		wdeath_day	wdeath_month	wdeath_year							
CHILDREN_INPUT	INDIVIDUAL	FAMREC	kidid	source_table	FIRST_NAME	name			Declared		birth_day	birth_month	birth_year							
CHILDREN_INPUT	INDIVIDUAL	FAMREC	kidid	source_table	LAST_NAME	lastname			Declared		birth_day	birth_month	birth_year							
CHILDREN_INPUT	INDIVIDUAL	FAMREC	kidid	source_table	BIRTH_DATE				Event		birth_day	birth_month	birth_year							
CHILDREN_INPUT	INDIVIDUAL	FAMREC	kidid	source_table	BIRTH_LOCATION	birthloc		birthloc_id	Event		birth_day	birth_month	birth_year							
PARENTS_INPUT	CONTEXT	FAMREC	parid	source_table	LEVEL		Union		Declared		marriage_day	marriage_month	marriage_year							
PLACES_INPUT	CONTEXT	FAMREC	placeid		LEVEL	level														Time_Invariant
PLACES_INPUT	CONTEXT	FAMREC	placeid		NAME	place														Time_Invariant

Back to page 63, 64, 67

Table 8. *Fields in IDS INDIV_INDIV, INDIV_CONTEXT and CONTEXT_CONTEXT Tables and IDS Transposer RELATIONSHIP Mapping File*

IDS INDIV_INDIV, INDIV_CONTEXT and CONTEXT_CONTEXT Tables		**IDS Transposer *RELATIONSHIP* Mapping File**	
Field name	**Description**	**Field name**	**Use**
Id	Primary key		Assigned automatically by IDS Transposer
		TableName	Name of data file for input. “PARENTS” is interpreted as “PARENTS.csv”
		RelationshipType	INDIV_INDIV, INDIV_CONTEXT, or CONTEXT_CONTEXT
Id_D	Identifier of the database or parts of the database from which the data are extracted.	*DatabaseID*	Short text describing the database
Id_I_1 **Id_I** **ID_C_1**	For INDIV_INDIV: Identifying number of the first individual in the relationship For INDIV_CONTEXT: Identifying number of an individual For CONTEXT_CONTEXT Identifying number of the less inclusive context in the relationship,	*FromEntityID*	Name of a column in the input file with an ID assigned to this individual or context.
Id_I_2 **Id_C** **ID_C_2**	For INDIV_INDIV: Identifying number of the second individual in the relationship For INDIV_CONTEXT: Identifying number of a context For CONTEXT_CONTEXT: Identifying number of the more inclusive context in the relationship	*ToEntityID*	Name of a column in the input file with an ID assigned to this individual or context.
Source	Specification of the source.	*Source*	Short text describing the source of the data within the database.
Relation	For INDIV_INDIV: Type of relationship of the first person to the second person. For example, person 1 is the “father” of person 2. For INDIV_CONTEXT: Description of the relationship between the context layers, e.g. neighborhood and municipality For CONTEXT_CONTEXT: The type of the relationship between individual and context, e.g. Boarder or Servant	*Relation*	Short text describing relationship between first ID and second ID to be assigned to every row
		RelationVariable	Name of a column in the input file with a relation value for each row

Table 8. *Fields in IDS INDIV_INDIV, INDIV_CONTEXT and CONTEXT_CONTEXT Tables and IDS Transposer RELATIONSHIP Mapping File (continued)*

IDS INDIV_INDIV, INDIV_CONTEXT and CONTEXT_CONTEXT Tables		**IDS Transposer *RELATIONSHIP* Mapping File**	
Field name	**Description**	**Field name**	**Use**
Date_type	Event, Reported, Declared, Assigned	*DateType*	An IDS date_type: Event, Reported, Declared, Assigned
Estimation	Type of estimate of the date or period (e.g. "Age_based," "Middling")	*DateEstimationType*	When this field is blank, IDS Transposer will compute a value using the date information provided ("Exact," "Month_year", "Year"). The user may also specify a column in the input file giving the estimation method for each row.
Day	Day number	*Day*	Name of a column in the input file
Month	Month number	*Month*	Name of a column in the input file
Year	Year number	*Year*	Name of a column in the input file
Start_day	Start day number	*StartDay*	Name of a column in the input file
Start_month	Start month number	*StartMonth*	Name of a column in the input file
Start_year	Start year number	*StartYear*	Name of a column in the input file
End_day	End day number	*EndDay*	Name of a column in the input file
End_month	End month number	*EndMonth*	Name of a column in the input file
End_year	End year number	*EndYear*	Name of a column in the input file
Missing	This field explains why a date or part of a date is missing	*DateMissingType*	Short text explaining why the date is missing, e.g. "Unavailable," "Time_Invariant"

Back to page 63, 66

Table 9a *INDIV_INDIV Table as Template for RELATIONSHIP File*

		Id_D	**Id_I_1**	**Id_I_2**	**Source**	**Relation**		**Date_type**	**Estimation**	**Day**	**Month**	**Year**	**Start_ day**	**Start_ month**	**Start_ year**	**End_ day**	**End_ month**	**End_ year**	**Missing**
Table-Name	*Relation-shipType*	*Databa-seID*	*FromEn-tityID*	*ToEn-tityID*	*Source*	*Relation*	*Relvar*	*DateType*	*Dateestima-tiontype*	*Day*	*Month*	*Year*	*Start-day*	*Start-month*	*Start-year*	*End-day*	*End-month*	*End-year*	*Datemissing-type*
		FAMREC		Marriage		Husband		Declared											
		FAMREC				Wife		Declared											
		FAMREC				Father		Declared											Time_Invariant
		FAMREC		Birth		Child		Declared											Time_Invariant
		FAMREC				Mother		Declared											Time_Invariant
		FAMRED				Child		Declared											Time_Invariant

Back to page 63, 66

Table 9b *INDIV_CONTEXT Table as Template for RELATIONSHIP File*

		Id_D	**Id_I**	**Id_C**	**Source**	**Relation**		**Date_ type**	**Estimation**	**Day**	**Month**	**Year**	**Start_ day**	**Start_ month**	**Start_ year**	**End_ day**	**End_ month**	**End_ year**	**Missing**
Table-name	*Relation-shiptype*	*Databa-seID*	*FromEn-tityID*	*ToEn-tityID*	*Source*	*Relation*	*Relvar*	*Datetype*	*Dateestima-tiontype*	*Day*	*Month*	*Year*	*Start-day*	*Start-month*	*Start-year*	*End-day*	*End-month*	*End-year*	*Datemissing-type*
		FAMREC		Marriage		Husband		Declared											
		FAMREC				Wife		Declared											
		FAMREC		Birth		Child		Declared											

Back to page 63, 66

Table 9c *CONTEXT_CONTEXT Table as Template for RELATIONSHIP File*

		Id_D	**Id_C_1**	**Id_C_2**	**Source**	**Relation**		**Date_type**	**Estimation**	**Day**	**Month**	**Year**	**Start_day**	**Start_month**	**Start_year**	**End_day**	**End_month**	**End_year**	**Missing**
Tablename	*Relationshiptype*	*DatabaseID*	*FromEntityID*	*ToentityID*	*Source*	*Relation*	*Relvar*	*Datetype*	*Dateestimationtype*	*Day*	*Month*	*Year*	*Startday*	*Startmonth*	*Startyear*	*Endday*	*Endmonth*	*Endyear*	*Datemissingtype*
		FAMREC		**Places**															Time_ Invariant
		FAMREC		**Marriage**		Union and Neighborhood		Declared											
		FAMREC		**Deaths**		Union and Neighborhood		Declared											
		FAMREC				Union and Neighborhood		Declared											
		FAMREC		**Birth**		Union and Neighborhood		Declared											

Back to page 63, 66

Table 10 *Mapping file: RELATIONSHIP*

TableName	***RelationshipType***	**DatabaseID**	***FromEntityID***	***ToEntityID***	**Source**	**Relation**	**RelationVariable**	**DateType**	**DateEstimationType**	**Day**	**Month**	**Year**	**StartDay**	**StartMonth**	**StartYear**	**EndDay**	**EndMonth**	**EndYear**	**DateMissingType**
PARENTS_INPUT	INDIV_INDIV	FAMREC	hid	wid	source_table	Husband		Declared		marriage_day	marriage_month	marriage_year							
PARENTS_INPUT	INDIV_INDIV	FAMREC	wid	hid	source_table	Wife		Declared		marriage_day	marriage_month	marriage_year							
CHILDREN_INPUT	INDIV_INDIV	FAMREC	dadid	kidid	source_table	Father		Declared											Time_Invariant
CHILDREN_INPUT	INDIV_INDIV	FAMREC	kidid	dadid	source_table	Child		Declared											Time_Invariant
CHILDREN_INPUT	INDIV_INDIV	FAMREC	momid	kidid	source_table	Mother		Declared											Time_Invariant
CHILDREN_INPUT	INDIV_INDIV	FAMREC	kidid	momid	source_table	Child		Declared											Time_Invariant
PARENTS_INPUT	INDIV_CONTEXT	FAMREC	hid	parid	source_table	Husband		Declared		marriage_day	marriage_month	marriage_year							
PARENTS_INPUT	INDIV_CONTEXT	FAMREC	wid	parid	source_table	Wife		Declared		marriage_day	marriage_month	marriage_year							
CHILDREN_INPUT	INDIV_CONTEXT	FAMREC	kidid	parid	source_table	Child		Declared		birth_day	birth_month	birth_year							
PLACES_INPUT	CONTEXT_CONTEXT	FAMREC	placeid	nested_in			relvar												Time_Invariant
PARENTS_INPUT	CONTEXT_CONTEXT	FAMREC	parid	marrloc_id	source_table	Union and Neighborhood		Declared		marriage_day	marriage_month	marriage_year							
PARENTS_INPUT	CONTEXT_CONTEXT	FAMREC	parid	hdeathloc_id	source_table	Union and Neighborhood		Declared		hdeath_day	hdeath_month	hdeath_year							
PARENTS_INPUT	CONTEXT_CONTEXT	FAMREC	parid	wdeathloc_id	source_table	Union and Neighborhood		Declared		wdeath_day	wdeath_month	wdeath_year							
CHILDREN_INPUT	CONTEXT_CONTEXT	FAMREC	parid	birthloc_id	source_table	Union and Neighborhood		Declared		birth_day	birth_month	birth_year							

Back to page 63, 66, 67

Table 11a *Example 2: MARRIAGES_INPUT*

parid	hfirst	hlast	wfirst	wlast	hocc	wocc	hid	wid	Marriage_day	Marriage_month	Marriage_year	marrloc	marrloc_id	source
1000	Franklin	Edwards	Nell	Kim	Salesperson	Eye Doctor	10001	10002	27	3	1786	Watts	104	Marriages
1001	Tristram	Hernandez	Marina	Kennedy	Police	Banker	10011	10012	24	5	1778	Tarzana	128	Marriages
1002	Bradford	Fisher	Cynthia	Vasquez	Historian	Film	10021	10022	27	6	1759	Glassell	101	Marriages
1003	Oliver	Peterson	Antonia	Porter	Designer	Maitre	10031	10032	29	8	1737	Montecito	137	Marriages
1004	Morgan	Lee	Adeline	Woods	English	Science	10041	10042	14	11	1807	Hollywood	166	Marriages

Back to page 63, 69

Table 11b *Example 2: DEATHS_INPUT*

idi	fname	lname	death_day	death_month	death_year	deathloc	deathloc_id	source
10001	Franklin	Edwards	25	11	1798	Watts	104	Deaths
10002	Nell	Kim	21	9	1810	Watts	104	Deaths
10011	Tristram	Hernandez	25	10	1796	Arleta	107	Deaths
10012	Marina	Kennedy	23	5	1817	Arleta	107	Deaths
10021	Bradford	Fisher	2	10	1804	Glassell	101	Deaths
10022	Cynthia	Vasquez	24	3	1822	Glassell	101	Deaths
10031	Oliver	Peterson	8	9	1806	Montecito	137	Deaths
10032	Antonia	Porter	20	3	1761	Montecito	137	Deaths
10041	Morgan	Lee	7	10	1819	Hollywood	166	Deaths
10042	Adeline	Woods	6	10	1840	Hollywood	166	Deaths

Back to page 63, 69

Table 11c *Example 2: BIRTHS_INPUT*

kidid	fname	lname	dadid	momid	birth_day	birth_month	birth_year	birthloc	birthloc_id	source
10001	Franklin	Edwards			28	3	1760	Glassell	101	Births
10002	Nell	Kim			16	1	1757	Boyle	136	Births
10011	Tristram	Hernandez			2	8	1752	Chatsworth	102	Births
10012	Marina	Kennedy			5	3	1753	Reynier	138	Births
10021	Bradford	Fisher			4	2	1744	Westwood	103	Births
10022	Cynthia	Vasquez			28	11	1743	Larchmont	145	Births
10031	Oliver	Peterson			7	2	1717	Watts	104	Births
10032	Antonia	Porter			14	8	1714	Brookside	139	Births
10041	Morgan	Lee			15	10	1773	Edendale	105	Births
10042	Adeline	Woods			24	11	1773	Cahuenga	129	Births
100001	Mona	Edwards	10002	10001	22	5	1788	Watts	104	Births
100101	Lora	Hernandez	10012	10011	27	7	1780	Tarzana	128	Births
100102	Cameron	Hernandez	10012	10011	5	9	1782	Tarzana	128	Births
100103	Roman	Hernandez	10012	10011	10	4	1784	Tarzana	128	Births
100104	Judith	Hernandez	10012	10011	25	2	1786	Tarzana	128	Births
100105	Roger	Hernandez	10012	10011	25	11	1788	Arleta	107	Births
100106	Stacy	Hernandez	10012	10011	2	11	1790	Arleta	107	Births
100107	Bertha	Hernandez	10012	10011	27	2	1792	Arleta	107	Births
100201	Vincent	Fisher	10022	10021	5	6	1761	Glassell	101	Births
100202	Darrell	Fisher	10022	10021	21	2	1763	Glassell	101	Births
100203	Lindon	Fisher	10022	10021	19	8	1765	Glassell	101	Births
100204	Noel	Fisher	10022	10021	5	1	1767	Glassell	101	Births

Back to page 69

Table 12 *Example 2: ENTITY Mapping File*

TableName	*EntityType*	*DataBaseID*	*EntityID*	*Source*	*Type*	*VariableName*	*Value*	*Value_ID_C*	*DateType*	*DateEstimationType*	*Day*	*Month*	*Year*	*Start Day*	*StartMonth*	*Start Year*	*EndDay*	*EndMonth*	*End Year*	*DateMissingType*
marriages_input	INDIVIDUAL	FAMREC	hid	source	FIRST_NAME	hfirst			Declared		marriage_day	marriage_month	marriage_year							
marriages_input	INDIVIDUAL	FAMREC	hid	source	LAST_NAME	hlast			Declared		marriage_day	marriage_month	marriage_year							
marriages_input	INDIVIDUAL	FAMREC	hid	source	SEX		Male													Time_Invariant
marriages_input	INDIVIDUAL	FAMREC	hid	source	OCCUPATION	hocc			Declared		marriage_day	marriage_month	marriage_year							
marriages_input	INDIVIDUAL	FAMREC	hid	source	MARRIAGE_DATE				Event		marriage_day	marriage_month	marriage_year							
marriages_input	INDIVIDUAL	FAMREC	hid	source	MARRIAGE_LOCATION	marrloc		marrloc_id	Declared		marriage_day	marriage_month	marriage_year							
marriages_input	INDIVIDUAL	FAMREC	wid	source	FIRST_NAME	wfirst			Declared		marriage_day	marriage_month	marriage_year							
marriages_input	INDIVIDUAL	FAMREC	wid	source	LAST_NAME	wlast			Declared		marriage_day	marriage_month	marriage_year							
marriages_input	INDIVIDUAL	FAMREC	wid	source	SEX		Female		Declared											Time_Invariant
marriages_input	INDIVIDUAL	FAMREC	wid	source	OCCUPATION	wocc			Declared		marriage_day	marriage_month	marriage_year							
marriages_input	INDIVIDUAL	FAMREC	wid	source	MARRIAGE_DATE				Event		marriage_day	marriage_month	marriage_year							

Back to page 69

Table 12 *Example 2: ENTITY Mapping File (continued)*

TableName	*EntityType*	*DataBaseID*	*EntityID*	*Source*	*Type*	*VariableName*	*Value*	*Value_ID_C*	*DateType*	*DateEstimationType*	*Day*	*Month*	*Year*	*Start Day*	*StartMonth*	*Start Year*	*EndDay*	*EndMonth*	*End Year*	*DateMissingType*
marriages_input	INDIVIDUAL	FAMREC	wid	source	MARRIAGE_LOCATION	marrloc		marrloc_id	Declared		marriage_day	marriage_month	marriage_year							
deaths_input	INDIVIDUAL	FAMREC	idi	source	FIRST_NAME	fname			Declared		death_day	death_month	death_year							
deaths_input	INDIVIDUAL	FAMREC	idi	source	LAST_NAME	lname			Declared		death_day	death_month	death_year							
deaths_input	INDIVIDUAL	FAMREC	idi	source	DEATH_DATE				Event		death_day	death_month	death_year							
deaths_input	INDIVIDUAL	FAMREC	idi	source	DEATH_LOCATION	deathloc		deathloc_id	Declared		death_day	death_month	death_year							
births_input	INDIVIDUAL	FAMREC	kidid	FIRST_NAME	fname			FIRST_NAME	Declared		birth_day	birth_month	birth_year							
births_input	INDIVIDUAL	FAMREC	kidid	LAST_NAME	lname			LAST_NAME	Declared		birth_day	birth_month	birth_year							
births_input	INDIVIDUAL	FAMREC	kidid	source	BIRTH_DATE				Event		birth_day	birth_month	birth_year							
births_input	INDIVIDUAL	FAMREC	kidid	source	BIRTH_LOCATION	birthloc		birthloc_id	Declared		birth_day	birth_month	birth_year							
places_input	CONTEXT	FAMREC	placeid		LEVEL	level														Time_Invariant
places_input	CONTEXT	FAMREC	placeid		NAME	place														Time_Invariant

Back to page 69

Table 13 *Example 2: RELATIONSHIP Mapping File*

TableName	***RelationshipType***	**DatabaseID**	***FromEntityID***	***ToEntityID***	**Source**	**Relation**	**RelationVariable**	**DateType**	**DateEstimationType**	**Day**	**Month**	**Year**	**Startday**	**Startmonth**	**Startyear**	**Endday**	**Endmonth**	**Endyear**	**DateMissingType**
marriages_input	INDIV_INDIV	FAMREC	hid	wid	source	Husband		Declared		marriage_day	marriage_month	marriage_year							
marriages_input	INDIV_INDIV	FAMREC	wid	hid	source	Wife		Declared		marriage_day	marriage_month	marriage_year							
births_input	INDIV_INDIV	FAMREC	dadid	kidid	source	Father		Declared		birth_day	birth_month	birth_year							
births_input	INDIV_INDIV	FAMREC	kidid	dadid	source	Child		Declared		birth_day	birth_month	birth_year							
births_input	INDIV_INDIV	FAMREC	momid	kidid	source	Mother		Declared		birth_day	birth_month	birth_year							
births_input	INDIV_INDIV	FAMREC	kidid	momid	source	Child		Declared		birth_day	birth_month	birth_year							
places_input	CONTEXT_CONTEXT	FAMREC	placeid	nested_in			relvar												
marriages_input	INDIV_CONTEXT	FAMREC	hid	marrloc_id	source			Declared		marriage_day	marriage_month	marriage_year							
marriages_input	INDIV_CONTEXT	FAMREC	wid	marrloc_id	source			Declared		marriage_day	marriage_month	marriage_year							
births_input	INDIV_CONTEXT	FAMREC	kidid	birthloc_id	source			Declared		birth_day	birth_month	birth_year							
deaths_input	INDIV_CONTEXT	FAMREC	idi	deathloc_id	source			Declared		death_day	death_month	death_year							

Back to page 69

Appendix B

R code to create PARENTS_INPUT, CHILDREN_INPUT, and PLACES_INPUT from PARENTS and CHILDREN

```
#read in original data tables
parents = read.csv(“parents.csv”)
children = read.csv(“children.csv”)

#create vector of location variables in parents.csv and empty vector for place names
locs = c(“hbirthloc”, “wbirthloc”, “marrloc”, “hdeathloc”, “wdeathloc”)
locations = NULL

#for each location variable, add place name to vector of names if not already included
for (j in 1:length(locs)){
  for (i in 1:nrow(parents)){
    if (as.character(parents[i,locs[j]]) %in% locations == FALSE){
      locations <- append(locations, as.character(parents[i,locs[j]]))
    }
  }
}

#add birth locations to vector of place names if not already included
for (i in 1:nrow(children)){
  if (as.character(children[i,"birthloc"]) %in% locations == FALSE){
    locations <- append(locations, as.character(children[i,"birthloc"]))
  }
}

#create place id codes and attach to names
placeid <- 101:(100+length(locations))
names(placeid) <- locations

#convert to dataset and write to csv
places_input <- data.frame(locations,placeid)
names(places_input) <- c(“place”,"placeid”)

#create individual identification variables for husbands and wives and source variable
parents[“hid”] <- parents[,"parid"]*10 + 1
parents[“wid”] <- parents[,"parid"]*10 + 2
parents[“source_table”] <-"PARENTS”

#convert dates to date format and extract components as separate variables
dates = c(“hbirth”, “wbirth”, “marriage”, “hdeath”, “wdeath”)
for (i in 1:length(dates)){
  datename = paste(dates[i], “date”, sep = “_”)
  dday = paste(dates[i], “day”, sep = “_”)
  dmonth = paste(dates[i], “month”, sep = “_”)
  dyear = paste(dates[i], “year”, sep=”_”)
  parents[datename] <- as.Date(parents[,dates[i]], format=»%Y-%m-%d»)
  parents[dday] <- as.numeric(format(parents[,datename], “%d”))
  parents[dmonth] <- as.numeric(format(parents[,datename], “%m”))
  parents[dyear] <- as.numeric(format(parents[,datename], “%Y”))
  parents[datename] <- NULL
}

#add context identification codes for location variables
locs = c(“hbirthloc”, “wbirthloc”, “marrloc”, “hdeathloc”, “wdeathloc”)
for (j in 1:length(locs)){
  locid = paste(locs[j], “id”, sep = “_”)
  parents[locid] <- 0
  for (i in 1:nrow(parents)){
    loc = as.character(parents[i,locs[j]])
    parents[i,locid] < placeid[loc]
  }
}

#write new dataset to csv
write.csv(parents, “parents_input.csv”, row.names = FALSE)

#create index variable for children and update it to count children born to each couple
children[“index”] <- 1
for (i in 2:nrow(children)){
  if (children[i,"parid"] == children[i-1,"parid"]){
    children[i,"index"] <- children[i-1,"index"] + 1
  }
}
#create identification code for children based on parid and index and source variable
children[“kidid”] <- children[,"parid"]*100 + children[,"index"]
children[“source_table”] <- “CHILDREN”

#add parents’ identification codes and father’s last name
children[“momid”] <- 0
children[“dadid”] <- 0

for (i in 1:nrow(children)){
  for (j in 1:nrow(parents)){
    if (children[i,"parid"] == parents[j,"parid"]){
      children[i,"momid"] <- parents[j,"wid"]
      children[i,"dadid"] <- parents[j,"hid"]
            children[i,"lastname"] <- parents[j,"hlast"]
    }
  }
}

#convert birth to date variable and extract components as separate variables
```

```
children[“bdate”] <- as.Date(children[,"birth"], format = “%Y-%m-%d”)
children[“birth_day”] <- as.numeric(format(children[,"bdate"], “%d”))
children[“birth_month”] <- as.numeric(format(children[,"bdate"], “%m”))
children[“birth_year”] <- as.numeric(format(children[,"bdate"], “%Y”))

#add context identification for birthloc
children[“birthloc_id”] <- NULL
for (i in 1:nrow(children)){
  loc = as.character(children[i,"birthloc"])
  children[i,"birthloc_id"] <- placeid[loc]
}

#create new dataset including old and new variables (drop index and bdate)
children_input <- subset(children, select = -c(index, bdate))
#write new dataset to csv
write.csv(children_input, “children_input.csv”, row.names = FALSE)
```