This document was supplied by the depositor and has been modified by the HDS

SN 6012 - Poor Jews' Temporary Shelter Database, 1896-1914

By Dr John Graham Smith (School of Historical Studies – University of Leicester), September 2008

1. **Outline of the database**

   Between 1886 and the outbreak of war in 1914, the Poor Jews' Temporary Shelter in London provided a transitory refuge for tens of thousands of migrants arriving from Eastern Europe in search of a better life in the West. The database contains a transcription of the personal and travel details of these migrants as recorded in the Shelter's surviving registers for the period 1896 to 1914, amounting to some 59,000 records. Besides their name, age, place of birth, occupation, marital status, and number of children (but unfortunately not for the most part their sex), the records also indicate the migrant's date of arrival and usually of departure, the amount of money they had with them, the place they were last from and sometimes the ship on which they arrived, and a London address at which they were staying or a destination for which they were bound elsewhere. For migrants travelling on to other countries, the ship on which they left is often given, particularly for those bound for South Africa, where the database makes clear the Shelter's important links with the Union and Castle shipping lines. There are also sometimes miscellaneous notes recording individual circumstances, such as a relative to whom the migrant was going. Derived columns have been added to the database to help identify migrants travelling together in family groups, and to link records in cases where a migrant's arrival was entered in both a main and a supplementary register.

2. **Development of the database**

   The database was the brainchild of Professor Aubrey Newman, who initiated a pilot project in 1987 with technical assistance from Cenka Inan, an IT trainee on work-experience placement. From 1988 it began to be developed in a more substantial way, under the historical direction of Professor Newman and the technical direction of Dr Graham Smith. They have together remained the project's joint directors ever since. The initial context was the launching at that time by the Leicester History Department of an ambitious Computing for Historians programme, driven forward by the enthusiastic advocacy of the then head of department, Professor Richard Bonney. This required all undergraduate historians at Leicester to take a sequence of compulsory computing courses, rooted primarily in database work and motivated by the belief that undergraduates should have first-hand exposure to the new kinds of research technique that were then starting to excite a growing number of historians. To that end a diverse assortment of databases was created, of which the Jews' Shelter database was the largest in terms of the student numbers it attracted. To build up these databases, and to familiarise students with the source materials, every student was required to undertake a quota of data input into the database to which they were assigned, and students' work culminated in their final year in the completion of an individual research project, based on statistical analysis of the data and written up as a historical dissertation. The programme continued along these lines until 2003, when it came to an end with the retirement of Graham Smith, who had run it since its inception.

   Although the Shelter database was initially developed primarily for teaching needs, its research potential was also obvious from the outset and this aspect received an important stimulus from interest it aroused at the University of Cape Town. One of the most striking early findings from the database was that a large proportion of the migrants it recorded - something like one third - were bound for South Africa. This made it a valuable resource for anyone interested in the early Jewish community there, and it attracted funding from the Kaplan Centre in Cape Town to help speed up data entry by employing postgraduate students as additional inputters. It also led the Kaplan Centre in 1997 to mount a partial online version of the database, designed for genealogical research (http://chrysalis.its.uct.ac.za/shelter/shelter.htm). The database has since been drawn on by two postgraduate theses and a number of research publications.

3. **Funding**

We are grateful to the Kaplan Centre at the University Cape Town for financial assistance that enabled us to employ some paid postgraduate inputters to supplement the work of our undergraduates. We are also indebted to the Department of History (now School of Historical Studies) at the University of Leicester, for its continuing support over many years.

4. **Source data and coverage**

The database is a transcription of the 18 surviving volumes of manuscript registers recording the 'inmates' of the Poor Jews' Temporary Shelter in London during the period May 1896 to July 1914. The registers are now housed in the London Metropolitan Archives (Ref. LMA/4184), although they remain the property of the Jews' Temporary Shelter. There are gaps in coverage as follows where registers have not survived: 1/11/1903-13/8/1904, 24/6/1905-28/11/1907, 28/11/1911-30/7/1913.

The registers consist of 12 main volumes and 6 supplementary volumes, the latter partly duplicating the contents of the main volumes but sometimes with slightly different information. There is no volume 7, as we discovered that the volume originally numbered 7 is in fact a further supplementary volume. The database has therefore renumbered this as volume 2b. For the purposes of the database, we have in some cases sub-divided volumes into decimal sections. These do not reflect any physical division of the volumes but are simply to accommodate the clerks' practice of re-starting their sequential numbering of the records on 1 November each year, in cases where the register spanned more than one of their working years. The following list shows the coverage of the registers as entered in the database.

Vol	Start Date	End Date	Start Ref	End Ref	Records
Main volumes					
01.0	1896.05.29	1896.12.31	1	* 3114.40391	2356
01.1	1897.01.01	1897.07.12	1	1351	1360
02.0	1897.07.12	1897.10.30	1352	2234	879
02.1	1897.11.01	1898.10.30	1	2100	2098
02.2	1898.11.01	1899.04.21	1	764	764
03.0	1899.04.21	1899.10.31	765	2694	1930
03.1	1899.11.01	1900.06.29	1	1823	1822
04.0	1900.06.29	1900.10.31	1824	2970	1163
04.1	1900.11.01	1901.10.31	1	2350	2349
05.0	1901.11.01	1902.10.30	1	2270	2287
05.1	1902.11.01	1902.12.19	1	1396	1461
06.0	1902.12.02	1903.10.30	1397	4492	3151
08.0	1904.08.14	1904.10.31	3840	4821	982
08.1	1904.10.31	1904.12.27	1	2185	2182
09.0	1904.12.27	1905.06.23	2186	5321	3127
10.0	1907.11.29	1908.10.29	321	1907	1586
10.1	1908.11.02	1909.09.28	1	1425	1425
11.0	1909.09.28	1909.10.31	1426	1547	122
11.1	1909.11.01	1910.09.05	1	2947	1768
12.0	1910.09.05	1910.10.31	2948	3327	389
12.1	1910.11.01	1911.10.31	1	2472	2407
12.2	1911.11.01	1911.11.27	1	207	207
13.0	1913.07.31	1913.10.31	1778	2357	586

Vol	Start Date	End Date	Start Ref	End Ref	Records
13.1	1913.11.01	1914.07.10	1	1153	1164
Supplementary volumes					
2a	1897.11.01	1898.12.13	1	4182	4227
2b	1897.11.01	1898.10.30	1	3642	3555
3a	1898.12.13	1899.12.17	1	3675	3975
4a	1899.12.01	1900.07.07	1	3669	3841
5a	1902.07.01	1902.10.10	1	3418	3867
6a	1903.01.01	1903.02.24	1	1917	1936

* The unusual reference number for the end of volume 01.0 reflects an error in numbering by the clerk. The format that we have used to record such errors will be explained below.

Some of the Shelter registers are now available online in digitised image format (pdf files) at www.movinghere.org.uk:

Vol. 1	May 1896-July 1897	Ref: LMA/4184/03/01/001. (58 files)
Vol. 2	July 1897-April 1899	Ref: LMA/4184/3/1/2. (65 files)
Vol. 3	April 1899-June 1900	Ref: LMA/4184/03/01/003. (58 files)
Vol. 2a	Nov 1897-Dec 1898	Ref: LMA/4184/03/01/018. (20 files)
Vol. 3a	Dec 1898-Dec 1899	Ref: LMA/4184/03/01/019. (60 files)
Vol. 4a,	Dec 1899-Jul 1900	Ref: LMA/4184/3/1/20. (61 files)
Vol. 10	Aug 1909 [one page only]	Ref: LMA/4184/3/1/10. (1 file)

3. Copyright

Copyright in the registers is owned by the Jews Temporary Shelter, 1-2 Endsleigh Street, London, WC1H 0DS. We are grateful to the Shelter for their permission to use the registers for the database, and for their agreement to the deposit of the database with the History Data Service (HDS). Copyright in the database itself is owned jointly by Professor Aubrey N. Newman and Dr John Graham Smith.

4. Software used

When the database was originally created in 1987, it was implemented using Ingres relational database software, running on the University's central VAX VMS system. Ingres was adopted simply because it was the only software available to us, but it proved well suited to the task and was to serve us well for over 12 years. In 1993, still using Ingres, the database was migrated to a UNIX platform, when the University replaced its old VAX system with Silicon Graphics Irix machines. Neither Ingres nor the platforms on which it ran could remotely be described as easy for the inexpert, so to facilitate student use we substantially customised the system. We wrote application programs in Ingres's own 4GL application language to drive input and query screens, while VAX DCL scripts and subsequently Korn Shell scripts were used to provide menus for easy access to the system, and to handle subsidiary tasks such as keyboard mappings and print queues. Access was originally via VT220 dumb terminals, and later via PCs running Kermit terminal emulation software when we needed to move our teaching into PC labs. By the late 1990s the system was looking distinctly old-fashioned, but we continued to appreciate the power and industrial-strength robustness of the system, important for our large-scale, multi-user needs. Eventually a change of software was forced upon us, when licensing considerations led the University to change its main database software from Ingres to SQL Server, and in 2001 the database was migrated to SQL Server 7. New front-end applications were now written in VBA and ADO, to provide access to the SQL Server back-end database via Microsoft Access. At the same time, we revised field names throughout our database tables to reflect Windows conventions. Thus, pers_id became PersId, and so on (some remnants

of the old names might still be encountered here and there *[This was undone by HDS due to HDS's file naming covnetions.]*). Throughout its 20-year history, the database has also seen gradual piecemeal evolution, with new fields added and old fields modified in response to changing needs and improving understanding of the data. At present the database continues to run on SQL Server, currently SQL Server 2000 with a migration to SQL Server 2005 in prospect.

5. Input methods

Some two-thirds of the input was carried out over a period of twelve years by some 300 conscripted undergraduates, as a compulsory part of their degree course. The remainder was mostly input by three postgraduate students, whose work was paid for by the funding from Cape Town. To bolster the motivation of our captive undergraduate workforce, marks were given for the quality of their work, contributing a small amount to their degree assessment. Despite this carrot, the effort and diligence shown by students varied widely, as did their expertise. We therefore had to give a good deal of attention to quality-control procedures, in an effort to safeguard the consistency and accuracy of the input. The undergraduates began by copying out their allocated set of register pages by hand on to transcription sheets. In the early years each student could tackle some 300 entries, but in the tighter constraints of a modular timetable this was reduced to under 100. The transcription sheets were then checked against the originals by the project's historical director (Aubrey Newman), who wrote in any amendments and returned the sheets to the students to be entered on to the computer. To guide students in their actual input, a detailed manual was provided setting out the rules and conventions for each field, while validation checks built into the input screen helped guard against the most obvious errors, such as invalid volume numbers and faulty date formats. Once the input was completed, a printout was checked by the two project directors in turn, first by Aubrey Newman for its historical accuracy and then by Graham Smith for its consistency with database rules. The marked printout was then returned to the student, for the indicated corrections to be made. These rather laborious procedures were deemed impracticable and unnecessary for the postgraduate students, who had previously completed the undergraduate course, so they were trusted to get on with the work with fairly minimal supervision.

6. Input quality

Despite the above procedures, the input that resulted was of very variable quality, partly because we did not have time for a final round of checking to ensure that the corrections we asked students to make were properly entered into the system. In the light of our experience, we cannot pretend that the use of undergraduate labour is an ideal way to build up a research database. The time spent in organising, instructing, chasing, checking, and marking, appears in retrospect disproportionate to the results achieved. In the long run it would have been quicker and more reliable just to do the input ourselves. On the other hand, without the help of our students the database would probably never have existed, and its construction was educationally important as a key element in our teaching programme. A great deal of further checking and correction has subsequently been made by Graham Smith, although there has not been time for the complete systematic review of the data that would ideally be needed. A good many errors therefore undoubtedly still remain, but we hope they do not seriously detract from the database's value.

7. General Input rules

Dates. All dates follow the pattern yyyy.mm.dd, e.g. 1899.08.23

Case. In general, input is entirely in lower case, except for (1) names of people, places, ships, institutions, etc., where the first letter is capitalised in the normal way, and (2) the Notes field, where the data usually takes the form of sentences which are therefore punctuated and capitalised as normal.

Abbreviations. Standard contractions and abbreviations are not normally given dots except in the case of shortened names, such as Abr. and Ch.

Blank fields. The database does not use null values. Text fields for which there is no data are simply left empty (interpreted by the database as a zero-length string). This is not possible in the case of numerical fields, which insist on having a number of some sort, so throughout the database we use -1 as a code to represent a 'missing value'. The code -1 is used, rather than 0, so as to differentiate between cases where the value is not known (represented by -1) and cases where it is known and is zero.

Spelling. Inputters were instructed to follow the spelling of places, occupations, ships, etc, as found in the source, and not try to correct or standardise the sometimes idiosyncratic spellings used by the clerks (e.g. 'cupertner' for 'carpenter'). Similarly, they were instructed to enter any abbreviations as written, and not try to expand or interpret them. These rules were not, however, always adhered to.

Interpolations and uncertain readings. The database uses curly brackets in the same way as square brackets in normal scholarly work, to indicate interpolations by the inputter. Square brackets were avoided for this purpose because of the system's technical use of square brackets as wildcard characters. Where an entry is uncertain, the database convention is to add '{?}' after it. Where a word or text string is illegible the convention is to indicate this by '{~}'. In practice, however, the difficult legibility of the Shelter registers means that readings are frequently open to some doubt, which would have made strict application of these conventions tedious in the extreme. They have therefore been only lightly used.

8. Database structure

The database is of simple basic structure, with data from the registers entered into just two tables, Persons and Moneys.

Persons – persons.tab holds the great bulk of the data, including each person's name, date of entry to the Shelter, age, place of birth, marital status, occupation, place last from, place gone to, and where applicable the ship on which they left.

Moneys – moneys.tab is a small subsidiary table, recording the money the person is declared to have had with them, sometimes in several different currencies, hence the need for a separate table.

These data input tables are supplemented by a number of small look-up tables, designed for use in SQL queries to regularise inconsistencies in the data, and to a lesser extent to help deal with errors in data input.

Occupations – occupations.tab allows for the translation of the highly diverse occupation designations written by clerks into standard occupation names or types

Places – places.tab similarly provides for the standardisation of place-name spellings and also for the expansion of place information, e.g. by identification of the country

Ships – ships.tab regularises faulty and inconsistent spellings of ship names

MCodes – m_codes.tab standardises currency codes, which again can be very variously written in the registers; it also allows for the (approximate) conversion of non-sterling amounts into sterling equivalents using exchange rates of the time.

The look-up tables - particularly Places - are somewhat rough and ready. They have proved useful for exploratory queries and broad analysis, and they were helpful for student projects, but they would need further development and refinement for detailed research.

9. Table descriptions

In the following table descriptions, the data types indicated are for SQL Server. For the most part their meanings will probably be clear to anyone familiar with database design, regardless of the software used. We should perhaps explain, however, that in the definitions of decimal data types, 'p' indicates the precision (i.e. the total number of digits stored) and 's' indicates the scale (i.e. the number of digits after the decimal point). For example, the data type that we have described as *decimal (9-byte, p:18, s:12)*,

used for the ref_num field in the Persons table, represents an 18-digit number, including 12 digits after the decimal point and occupying storage space of 9 bytes.

Persons table – persons.tab

Table to record all the personal and travel details about each migrant, apart from the money they had with them which is instead recorded in the separate Moneys table. (58,966 rows)

Field name *data type*	**Content**
inid_p *varchar(6)*	Two-part code to identify the inputter, e.g. se89. The numerical part denotes the academic session in which they started their course.
chk_p *varchar(6)*	A check field, used for temporary working purposes only.
pers_id *integer [4-byte]*	A unique identification number for the record, assigned automatically.
fam_id *integer [4-byte]*	A derived column, containing an identification number for the family group to which the person belonged. If the person was travelling alone, this is the same as the pers_id. If the person was travelling with other family members, the pers_id of the first member of the group listed is assigned as the fam_id to each member of the group. fam_id values were assigned after the completion of the input by running an SQL script. The algorithm used depends on the assumption that migrants with the same surname, entering the Shelter on the same date, coming from the same place, going to the same place, and listed consecutively in the register, constituted a family group. The assumption might not always be correct but it is a useful working hypothesis.
gp *smallint [2-byte]*	A derived column, indicating the size of the family group in which the person was travelling. Calculated by running an SQL script on the fam_id column. Migrants travelling alone have a group size of one.
match_1 *integer [4-byte]*	A derived column, to link cases where a migrant's arrival at the Shelter is recorded both in a main register and in one of the 'a' series supplementary registers (2a, 3a, 4a, 5a and 6a). The value in the match_1 column is the pers_id value of the other record to which the present record can be linked. It is useful to identify such links because the information given in the supplementary register can sometimes differ slightly from that in the main register, e.g. by giving full forenames instead of simply initials. The values in this column were assigned by running an SQL script after the completion of data input. The algorithm used is based on a comparison of surname, forename, date of entry and age, and it takes account of variant spellings by including pragmatically-defined partial matches. A first sweep of the data identified matches where the following set of criteria was met: (1) Length of surname in main volume differs by no more than one character from length of surname in supplementary volume. (2) The first two letters and last two letters of the surname in one volume match those in the other OR the surname in the main volume is at least 4 characters long and all but the last two characters match the corresponding characters in the supplementary volume OR the surname in the main volume is at least 5 characters long and all but the last three characters match the corresponding characters in the supplementary volume. (3) The first character of the forename in one volume matches that in the other. (4) The age in one volume matches that in the other. (5) The date of entry in one volume differs by no more than one day from that in the other. In cases where the first sweep of the data found no match, a second sweep was made using the same set of criteria except for the requirement for a match on the first character of the forename. This second sweep served to identify cases where initials are easily confused (N/U, J/I, etc.).

Field name *data type*	**Content**
match_2 *integer [4-byte]*	A similar derived column to link cases where a migrant's arrival is recorded both in a main register and in supplementary register 2b. In some cases the database has three records of a migrant's arrival: in a main register, in an 'a' series supplementary register, and in supplementary register 2b.
r_stat *smallint [2-byte]*	A code to indicate the status of the row: 1 = A normal register entry 2 = An inferred entry, where we have assumed the presence of an accompanying child (see below, fields ref_num and family). -2 = A row numbered in the register by the clerk but left blank -3 = A row deleted by the clerk -4 = A row for which we have no data because of a gap in the microfilm from which we worked.
vol *varchar(5)*	Volume number (as listed above in section 'Source data and coverage')
ref_num *decimal (9-byte, p:18, s:12)*	A sequential reference number for the migrant, as written in the register by the clerk (or added by ourselves if the clerk left the records unnumbered). Occasionally clerks slipped up and got their numbering wrong, so that the numbers do not follow in correct sequence. In a case like this we have entered a compound reference number, made up of the last correct number, then a decimal point, then the erroneous number that the clerk actually wrote, and finally a 1 (one) to mark the end of the number. For example, if after the correct number 263 the clerk absent-mindedly numbered the following entries 244, 245, 246, before recovering himself and going on to 264, ref_num values would be entered as follows: 263 263.2441 263.2451 263.2461 264 The purpose of this scheme is to preserve the numbers actually written while at the same time enabling us to keep the records in correct order in the database. In cases where the number to be added after the decimal point is less than 100, it is padded with zeros to make three digits, e.g. 9, 9.0081, 9.0091, 10, 11 … The pattern we have followed is not entirely consistent, however, as in later input we encountered cases where the clerk's error involved a four-digit number, so the number after the decimal point is four digits (plus one). Another problem sometimes encountered is where the clerk omits to give the odd few records any numbers at all. In such cases, we supply decimal-point numbers starting at 1, again padded with zeros to make three digits (but this time omitting an end-marker '1'). For example: 749 749.001 749.002 750 A similar method is used in cases where we have added records for people (e.g. wife or children) who are described in the register as accompanying a person but who have not been given register entries of their own. For example, if the person with ref_num 823 is described as having two children with him, they would be given database entries with ref_num values 823.001 and 823.002.
date_ent *varchar(11)*	The date of entry to the Shelter, e.g. 1897.02.26

Field name *data type*	**Content**
forename *varchar(20)*	The person's forename(s) or initial(s) as given by the clerk. Dots are omitted after initials but included after abbreviated names, such as Sch. or Abr. Where there is more than one initial, they are spaced, e.g. C A, not CA. Where a title is given, it is added after the forename and preceded by a comma and a space. For example: Libe, Mrs If only a title is given, it is still preceded by a comma and space: , Mrs If the title accompanies an abbreviated name, there is both a dot and a comma: Sch., Mrs
surname *varchar(20)*	Surname as written by the clerk, except that in the rare cases where the clerk wrote the name with an umlaut (or other accent) the umlaut is ignored.
jewish *varchar(5)*	Field not used. It was intended to be used to differentiate between Jewish and non-Jewish migrants.
sex *varchar(3)*	Unfortunately, the registers do not record a migrant's sex. Where we could clearly infer it from a title, such as Mrs, Miss or Master, we have entered 'f' or 'm'. In almost all other cases we have left the sex blank. We have not attempted to infer it from the forename or occupation, with very rare exceptions where it was unambiguous (e.g. for a tailoress).
age *decimal (5-byte, p:9, s:4)*	Age in years. Ages given in the register in months are converted to decimal form. Where no age is given, the database uses -1 as a 'missing value' code. If the register gives an age simply as 'under 1', it is entered as -2.
Place_born *varchar(20)*	Place of birth, following the spelling in the register.
mar_stat *varchar(5)*	Marital status, using the codes 'm', 's' and 'w'
num_child *smallint [2-byte]*	Number of children as written in the register column 'No. of Childn'. We do not enter any figure here if the clerk has not written a figure in the register, even if the register lists children accompanying the person. Where no figure is entered, the database has the 'missing value' code -1.
family *varchar(10)*	In the Family field we enter anything the clerk has written in the register column headed 'Where the family now are', typically 'here' or 'at home'. 'Here' is taken to mean that the children indicated in the register under 'No. of Childn' are accompanying the person. If the register has not given these children entries of their own, we add entries for them to the database, with a ref_num value composed as described above in the description of the field ref_num. In these cases we assume that the travel information is the same for the children as for the parent (last_from, place_to, etc.). We have also often assumed that the place of birth is the same as for the parent, but we have not been consistent in this.
last_from *varchar(20*)	The place the register indicates the person to be 'last from', typically their Continental port of departure.
in_via *varchar(15)*	The person's port of entry to England, where the register gives this (usually under the heading 'Where last from')..
ship_in *varchar(25)*	The ship on which the person arrived, sometimes indicated along with their Continental port of departure in the register column headed 'Where last from'.

Field name *data type*	**Content**
occup *varchar(20)*	The person's occupation, entered in lowercase but otherwise as written in the register. The clerks were often highly inconsistent in the way they wrote occupations. For example, a boot maker might be written as boot m, boot m., boot ma, boot mk, boot mk., boot mkr, boot'm, bootmaker, bootmkr, bootmkr., and so on. Inputters were instructed not to attempt to interpret, modernise, or standardise such descriptions. Instead a separate look-up table is used to regularise these inconsistencies (Occupations table)
recommend *varchar(20)*	A field to indicate any person or institution mentioned in the register entry as having recommended the person. Very rarely used.
stay *decimal (5-byte, p:9, s:4)*	The person's length of stay, as indicated in the register column headed 'How many days in the institution'. Often left blank by the clerks in which case we do not enter a figure in the database: we do not attempt to calculate it from the person's date of entry and date of leaving.
date_left *varchar(11)*	Date the person left, as written in the register column headed 'When left it' or in the column headed 'General remarks on their leaving institution'. In cases where no leaving date is given, we have not sought to calculate it from the date of entry and length of stay.
date_sail *varchar(11)*	Column not used. Originally intended to differentiate between dates written in the register column 'When left it' and dates written in the 'General remarks' column, usually after the name of the ship on which they were leaving. However, we found that the distinction seems insignificant, the date apparently being written in one place or the other simply at the whim of the clerk.
a_flag *varchar(12)*	An 'address flag' field, to record the annotation 'N.A', 'N.a', etc. (variously written), that clerks sometimes wrote in the 'Where gone to' column of the register. We originally wrongly interpreted this as meaning 'Native abode', and took it to mean that the person was returning to their native town, until we found in due course that it evidently means 'No address'.
num_to *varchar(8)*	The house number in cases where the register gives a street address as the place the person has gone to.
PlaceTo *varchar(30)*	The place the person has gone to (e.g. Africa), or the street name where a street address is given. Spellings follow those in the register, except that inputters were instructed to use the following standard abbreviations in order to minimise the degree of inconsistency: Ave=Avenue, Bldgs=Buildings, Ct=Court, La=Lane, Pl=Place, Rd=Road, Sq=Square, St=Street, Ter=Terrace, Yd=Yard.
num_to_2 *varchar(8)*	The registers sometimes indicate a second destination (usually under 'General remarks on their leaving'), presumably the place the person is going on to after their initial destination. The num_to_2 field holds any house number indicated for this second destination.
place_to_2 *varchar(30)*	The rest of the place indicated as the person's second destination.
ship_out *varchar(25)*	The name of the ship on which the person left, where this is given (usually under 'General remarks on their leaving'). Any prefixes or suffixes are omitted (e.g. ss, s/s, U.L., etc).
pass_paid *varchar(20)*	If the clerk has written 'P.Pd' or similar, meaning 'passage paid', this is indicated with a 'y' in the PassPaid field.
agent *varchar(20)*	If the register mentions the name of an agent, this is entered in the agent field.
joining *varchar(20)*	If the register indicates a family relation whom we can presume the person to be joining or staying with, the relation is entered in the Joining field (as written by the clerk). For example: 'cousin', 'B. in L.', etc.

Field name *data type*	**Content**
notes *varchar(200)*	A catch-all field, to record any further information for which there is no column in the database table, or any comment the inputter wishes to make. Any text transcribed exactly from the register is enclosed in double quotation marks so as to differentiate it from inputters' summaries, paraphrases or comments.

Moneys table – moneys.tab

Table to record the amounts of money people had with them, usually written in the register under the heading 'General remarks … on entering institution'. (59,161 rows)

Field name *data type*	**Content**
inid_m *varchar(6)*	Code to identify the inputter (as in Persons table).
chk_m *varchar(6)*	A check field, used for temporary working purposes only.
pers_id *integer [4-byte]*	A unique identification number that correlates the Moneys record with the corresponding record in the Persons table.
r_money *varchar(10)*	The money amount as recorded in the register (r_money stands for Recorded Money). Sterling amounts are entered in the form 2.12.06, signifying in this example 2 pounds, 12 shillings and 6 pence. If the register indicates several sums, in different currencies, each is entered in the Moneys table as a separate record. Where the person is recorded to have had no money (usually written as 'N.M.' in the register), this is entered as 'nm'. The code 'nm' is entered only if the register explicitly indicates that the person had no money: if the register says nothing about money, the r_money field is left blank.
m_type *varchar(12)*	The currency, as written in the register (e.g. Rs, M, ag, DOL). For sterling amounts, the clerks usually do not state the currency and for these the database uses the standard abbreviation stg.
m_code *varchar(12)*	A derived column, which translates the m_type into a standard code (via the look-up table m_codes).
d_money *decimal (5-byte, p:9, s:4)*	A derived column, expressing the money amount in decimal form.
s_money *decimal (5-byte, p:9, s:4)*	A derived column, converting foreign money to a sterling equivalent (via the look-up table m_codes), using the exchange rates prevailing at the time. Obviously this can only be very approximate but it sufficed for teaching purposes.

Occupations table – occupations.tab

A look-up table, used via SQL queries for converting the very diverse occupation designations written by clerks into standardised occupation names. (1,213 rows)

Field name *data type*	**Content**
inid_o *varchar(6)*	Code to identify the inputter. Only patchily used and of no continuing relevance.
chk_o *varchar(6)*	A check field, used for temporary working purposes only.

Field name *data type*	**Content**
occup *varchar(20)*	The occupation as written by the clerk, e.g. boot m, boot mkr, bootmaker, etc. Used as the 'key' column for SQL joins with the occup column in the Persons table.
occup_type *varchar(20)*	A standardised occupation name, e.g. 'boot maker'

Places table – places.tab

A look up table, used via SQL queries to regularise the very diverse spellings (and misspellings) of place names in the Persons table, and to identify the country. (9,009 rows)

Field name *data type*	**Content**
inid_pl *varchar(6)*	Code to identify the inputter. Only patchily used and of no continuing relevance.
chk_pl *varchar(6)*	A check field, used for temporary working purposes only.
place *varchar(30)*	The place name as entered in the place_born, last_from, Iin_via, place_to or place_to_2 column in the Persons table. Used as the 'key' column for joins with whichever place column in the Persons table one is interested in.
street *varchar(30)*	A regularised street address.
town *varchar(20)*	A regularised name for the town or locality. In the case of birthplaces, it is often unclear whether the place indicated in the register refers to the town or to the Russian government district that takes its name from the town (e.g. Kovno, Grodno, Minsk, Kiev). In practice, therefore, when working with birthplaces this column can mean either the town or the government district.
dist *varchar(4)*	Field intended to hold the London district in cases where clerks added this to a street address (e.g. 'E', 'W'). Little used.
region *varchar(20)*	Field intended to identify the region but largely left unused.
country *varchar(20)*	Field to identify the country. For the purposes of student project work, this was found to be particularly useful for birthplaces. Where the place could be identified, the country entered here is usually that given in the 11th edition Encyclopaedia Britannica.

Ships table – ships -tab

A look up table, used via SQL queries to regularise the various spellings (and misspellings) of ship names in the Persons table. (478 rows)

Field name *data type*	**Content**
inid_s *varchar(6)*	Code to identify the inputter. Only patchily used and of no continuing relevance.
chk_s *varchar(6)*	A check field, used for temporary working purposes only.
ship_input *varchar(25)*	The ship name as entered in the ship_in or ship_out column of the Persons table. Used as the 'key' column for joins with whichever of the two columns in the Persons table one is interested in.
ship *varchar(25)*	Regularised ship name.

Field name *data type*	Content
line *varchar(25)*	Field intended to identify the shipping line but largely left unused.

MCodes table – m_codes.tab

A look-up table used for populating the m_code and s_money columns in the Moneys table, via SQL joins. (195 rows)

Field name *data type*	Content
m_type *varchar(12)*	The currency type as entered in the m_type column of the Moneys table. Used as the 'key' field for SQL joins.
m_code *varchar(12)*	A standardised code for the currency type.
m_equiv *decimal (5-byte, p:9, s:6)*	Contemporary exchange rate, used for converting foreign moneys into sterling equivalents.
M_descrip *varchar(20)*	A fuller standardised description of the currency.

10. Bibliography

Newman, A. & Smith, J. G., 'The Poor Jews' Temporary Shelter: the development of a database on Jewish migration, 1896-1914'. In R. Bonney (ed.), University of Leicester, Department of History Occasional Papers Series, 2000. [Wrongly dated 2000 on the Web page; should be 1996] http://www.le.ac.uk/hi/teaching/papers/jewspap.html

Newman, A. N., Evans, N. J., Smith, J. G. & Issroff,, S. W. (eds.), *Jewish migration to South Africa: the records of the Poor Jews' Temporary Shelter, 1885-1914*, JPSA (Jewish Publications - South Africa), Isaac and Jessie Kaplan Centre, University of Cape Town, 2006. ISBN: 0-7992-2315-8 [A provisional checklist with historical introduction and substantial bibliography]

Newman, A., 'Trains and shelters and ships'. In R. Bonney (ed.), University of Leicester, Department of History Occasional Papers Series, 2000.
http:// http://www.le.ac.uk/hi/teaching/papers/newman1.htm

Newman, A., 'The Union Castle Line and emigration from Eastern Europe to South Africa'. In R. Bonney (ed.), University of Leicester, Department of History Occasional Papers Series, 2000.
http://www.le.ac.uk/hi/teaching/papers/newman2.html

Newman, A., 'The Poor Jews' Temporary Shelter: an episode in migration studies', *Jewish Historical Studies*, 40 (2005), 141-155

Unpublished theses:

Barker, C. L. 'Jewish migration to South Africa and the Poor Jews' Temporary Shelter, London, 1880-1914' (M. Phil. thesis, University of Leicester, 1998)

Evans, N. J. 'Aliens en route: European transmigration via Britain, 1836-1914' (Ph.D. thesis, University of Hull, 2006)