1. Introduction

1.1 Purpose of VPLX

VPLX is a Fortran program for the estimation of variances from complex sample surveys through replication methods. In general, replication methods obtain an estimate of sampling variance by carrying out the estimation for the full sample on a series of subsamples of the data, and the program is specifically designed for this purpose.

Most Census Bureau and other government sample surveys, as well as many other national surveys, employ complex sample designs. One consequence is the typical use of weighted data in estimation. Many widely available statistical systems, such as SAS, SPSS, and others, are able to use the survey weights provided on public use files to reproduce the official estimates. Typically, however, the variance estimates provided by these systems do not consider the effect of the complex design on the reliability of the estimates. In some situations, variance estimates based on an assumption of simple random sampling may be roughly corrected by rules of thumb. For many purposes, including the Census Bureau's own research, more accurate variance estimates are required. VPLX and some other systems, such as SUDAAN and WesVarPC, now provide a means to do so for a variety of designs and estimation problems.

A VPLX application is organized into a series of steps. Parallels between VPLX and the SAS system are possible here. For example, a SAS user may assemble a SAS data set from raw input data with the DATA step and then run one or more of the available SAS procedures. Typically, the SAS data set contains data at the level of the individual observation. SAS steps may be run independently of each other because the information about variables is communicated between steps as metadata included with the SAS data sets. Similarly, VPLX steps are almost completely independent and communicate with each other through the contents of *VPLX files*.

The CREATE step prepares a VPLX file from a file of individual observations. Unlike a SAS data set, however, the VPLX file contains estimated totals for the full sample and for the replicate samples corresponding to the replication method, rather than data for individual observations. Virtually any VPLX application incorporates one or more CREATE steps or uses a VPLX file that depended directly or indirectly upon a previous CREATE step. Thus, the CREATE step is a fundamental component of the system. This and the following three chapters describe basic features of CREATE.

Once produced by the CREATE step, the VPLX file may be input directly into the DISPLAY step, which can display estimates, standard errors, covariances, correlations, and t-tests based on the

VPLX file. A number of examples in this and the next chapters include accompanying DISPLAY steps. The TRANSFORM step can read a VPLX file, derive new variables that are functions of sample totals and include them on a new VPLX file, which in turn can be input to DISPLAY.

A VPLX file has a complex structure to organize the data required for variance estimation through replication. The VPLX file holds both data and *metadata*, characteristics of the data such as choice of replication method, variable names, labels, and other attributes. The design of the VPLX file has undergone far fewer changes than the program, although the addition of new features may force a redesign of the VPLX file within the next few years (*1*). An attempt will be made to maintain upward compatibility in implementing this revision. As of October 1997, the current file design is 92.03, used since 1992, but the program still accepts its predecessor, 90.04, as input to most steps.

In addition to VPLX files, VPLX applications often require reading from and writing to several different files. Two files are particularly important: the character input file containing the VPLX language commands prepared by the user, which will generally be called the *command file*; and an output *print file*, to which VPLX generally echoes commands and reports results. In practice, of course, users may usually prefer to inspect the print file in an editor rather than actually printing it.

1.2 The Standard Order for the CREATE Step

The CREATE step may be understood as composed of five key components.

```
CREATE in = ...  out = ;

     Input section  - read data from the in = file and
       possibly others

     Replication specification section  - provide commands to
       define the replication method

     Recode section - revise existing variables or create new
       ones with arithmetic and logical statements.

     Categorization section - define categorical and class
       variables

     Output section - specify how output variables are to be
       cross-classified and for what universes.
```

Exhibit 1.1 The five sections of a VPLX CREATE step in the *standard order.*

When these components are organized according to exhibit 1.1, the CREATE step is in the *standard order*. The CREATE step does not require the standard order, but this organization carries a number of advantages. It divides the step into a number of separate tasks, and it is easier to understand specific VPLX statements in the context of these tasks. Secondly, the consequences of VPLX statements are much clearer when the CREATE step is in the standard order. When a VPLX CREATE step departs from the standard order and produces unexpected results, reorganizing the code into standard order may clarify the problem. The documentation will focus almost exclusively on the use and interpretation of CREATE statements in the standard order.

Exhibit 1.2 provides a simple example of a command file with the CREATE step in the standard order.

```
!  i1-1.crd

scratch1  temp1.tmp;          ! Optional naming of scratch files.
scratch2  temp2.tmp;
scratch3  temp3.tmp;
scratch4  temp4.tmp;
scratch5  temp5.tmp;

/*       This first example illustrates a simple VPLX run.
         Except for these general comments, other comments
         appear to the right, following an exclamation mark.

 The input data are the six records of i1-1.dat:
 5 7 1 1 1
 6 8 1 2 1
 5 2 1 3 2
 4 1 1 4 2
 8 4 1 5 3
 8 2 1 6 3
*/

create  in = i1-1.dat  out = vplx1.vpl ;  ! Beginning of CREATE step

  input  rooms persons  weight cluster    ! input statement to read
          stratum /format (5F2.0) ;       ! data  ("input section")

  replication method simple jackknife;    ! ("replication specification
  replicate number cluster;               ! section")

  room_ratio = rooms/persons;             ! ("recode section")
  if persons > rooms then;                ! Note: overcrowded will be
     overcrowded = 1;                     ! 0 or 1.
  end if;
  print rooms persons room_ratio
    overcrowded / option nprint = 3;

  class overcrowded (1/0) 'Overcrowded'   ! Definition of a class
                      'Not overcrowded';  ! variable
                                          ! ("categorization section")

  block rooms persons                     ! Definition of a block
        room_ratio / class overcrowded;   ! ("output section")
```

```
display                                     ! Beginning of DISPLAY step

list N(1) rooms persons room_ratio          ! specification of
     total (rooms persons room_ratio)       ! display
     / class total overcrowded (1)
```

Exhibit 1.2 An example command file. Several lines of comments appear between "/*" and "*/"; other comments appear to the right following an exclamation mark, "!." The example illustrates the standard order. A brief DISPLAY step follows, which uses the output VPLX file from the CREATE step to produce its results.

The command file was submittted to VPLX on a PC under Windows 95 with the following command in a DOS window:

```
vplx < i1-1.crd > i1-1.lis
```

and the resulting print file, i1-1.lis, was

```
                              VPLX  -  Version 1997.12
!  i1-1.crd

scratch1  temp1.tmp;          ! Optional naming of scratch files.

scratch2  temp2.tmp;

scratch3  temp3.tmp;

scratch4  temp4.tmp;

scratch5  temp5.tmp;

/*       This first example illustrates a simple VPLX run.
         Except for these general comments, other comments
         appear to the right, following an exclamation mark.

 The input data are the six records of i1-1.dat:
 5 7 1 1 1
 6 8 1 2 1
 5 2 1 3 2
 4 1 1 4 2
 8 4 1 5 3
 8 2 1 6 3
*/


create  in = i1-1.dat  out = vplx1.vpl ;  ! Beginning of CREATE step


  input  rooms persons  weight cluster    ! input statement to read
          stratum /format (5F2.0) ;       ! data  ("input section")

  replication method simple jackknife;    ! ("replication specification
```

```
  replicate number cluster;               ! section")

  room_ratio = rooms/persons;             ! ("recode section")
  if persons > rooms then;                ! Note: overcrowded will be
     overcrowded = 1;                     ! 0 or 1.
  end if;
  print rooms persons room_ratio
    overcrowded / option nprint = 3;

  class overcrowded (1/0) 'Overcrowded'   ! Definition of a class
                     'Not overcrowded';   ! variable
                                          ! ("categorization section")

  block rooms persons                     ! Definition of a block
        room_ratio / class overcrowded;   ! ("output section")


    Size of block   1  =              8

    Total size of tally matrix =      8

**** End of CREATE specification/beginning of execution
PRINT request    1
     rooms               5.0000          persons             7.0000
     room_ratio           .7143          overcrowded         1.0000
PRINT request    1
     rooms               6.0000          persons             8.0000
     room_ratio           .7500          overcrowded         1.0000
PRINT request    1
     rooms               5.0000          persons             2.0000
     room_ratio          2.5000          overcrowded          .0000

    End of primary input file after obs #       6


display                                   ! Beginning of DISPLAY step


list N(1) rooms persons room_ratio        ! specification of
     total (rooms persons room_ratio)     ! display
     / class total overcrowded (1)

                                                    Estimate      Standard error

Sample N (wd) for block   1                          6.0000              .0000

rooms                  :  MEAN                       6.0000              .6831

persons                :  MEAN                       4.0000             1.1832

room_ratio             :  MEAN                       2.3274              .6006

rooms                  :  TOTAL                     36.0000             4.0988

persons                :  TOTAL                     24.0000             7.0993

room_ratio             :  TOTAL                     13.9643             3.6037
```

```
    overcrowded             :   Overcrowded

                                                    Estimate      Standard error

Sample N (wtd) for block   1                          2.0000              1.2649

rooms                   :  MEAN                       5.5000               .6455

persons                 :  MEAN                       7.5000               .6455

room_ratio              :  MEAN                        .7321               .0231

rooms                   :  TOTAL                     11.0000              7.0000

persons                 :  TOTAL                     15.0000              9.5184

room_ratio              :  TOTAL                      1.4643               .9265


Use of double precision matrix:      70 out of 1000000
Stop - Program terminated.
```

Exhibit 1.3 An example print file, i1-1.lis. The print file echoes all the statements from the command file, i1-1.crd. After listing the CREATE step specification, the print file provides a summary of the size of the problem and results from the PRINT statement, shown in italics here. VPLX then echoes the DISPLAY step specification and provides the results.

Exhibits 1.2 and 1.3 illustrate how the print file echoes the command file. Subsequent examples will show only the print file.

More substantively, the example illustrates each of the sections of the standard order:

- Input section - a single INPUT statement specifies the input variables and the associated format. These data are read from the input data file, i1-1.dat.

- Replication method specification section - two statements specify the replication method and that the variable CLUSTER is to be used in defining the replicates.

- Recode section - statements define two new variables and specify printing of some cases to the print file.

- Classification section - the specification for OVERCROWDED as a class variable allows it to cross-classify other variables.

- Output section - a BLOCK statement specifies the variables to be included in the VPLX file and how they are cross-classified.

On occasion, statements may be omitted for one or more sections except the first, as the following example illustrates.

```
!  i1-2.crd

!  i1-2.crd is a minimalist version of i1-1.crd
!  Using the same input data, it relies on defaults as
!  much as possible.


scratch1  temp1.tmp;          ! Optional naming of scratch files.

scratch2  temp2.tmp;

scratch3  temp3.tmp;

scratch4  temp4.tmp;

scratch5  temp5.tmp;


create  in = i1-1.dat  out = vplx1.vpl ;  ! Beginning of CREATE step


  input  rooms persons  weight cluster    ! input statement to read
          stratum /format (5F2.0) ;       ! data  ("input section")

! Without a replication specification section, the appearance of
! cluster and stratum among the input variables will result in the
! default selection of the stratified jackknife.

! Without a recode section, the only variables will be those
! read in.

! Without categorization and output sections, a single block
! will be created with rooms and persons.


    Stratified jackknife replication assumed

    Size of block   1  =             3

    Total size of tally matrix =     3

**** End of CREATE specification/beginning of execution

    End of primary input file after obs #       6

         3 strata observed on incoming file


display                                    ! Beginning of DISPLAY step


list N(1) rooms persons                    ! Display specification
     total (rooms persons )                ! is necessarily limited
                                           ! to available variables
```

```
                                                    Estimate      Standard error

Sample N (wtd) for block   1                          6.0000              .0000
rooms                   :  MEAN                       6.0000              .2357
persons                 :  MEAN                       4.0000              .4082
rooms                   :  TOTAL                     36.0000             1.4142
persons                 :  TOTAL                     24.0000             2.4495
```

Exhibit 1.4 Example print file, i1-2.lis, illustrating the effect of defaults when no statements are given for all but the input section of the standard order. In this example, default selections exist for all of the remaining sections. For the sake of brevity, the first line, last two lines, and some blank lines of the print file have been omitted. The italicized report from the CREATE step and the output from the DISPLAY step are not part of the command file, i1-2.crd.

Thus, the task of documentation includes indicating the consequences when one or more of the standard sections is missing.

This documentation is organized according to the sections of the standard order, although in a different sequence.

- Chapter 2 describes the most frequently used statements of the recode section, including arithmetic and logical statements. The syntax of these statements is similar and in places identical to the syntax of SAS or Fortran. Although this chapter focuses on the middle rather than the beginning of the CREATE step, it is helpful to do so, since the statements of the recode section are the most similar to other languages. Furthermore, recode statements must sometimes be included in the input section, for example, to read conditionally from a second input file, or in the replication method specification section, when specification of the replication method requires that some input variables be redefined.

- Chapter 3 describes the statements of the classification and output sections. These sections are generally related to each other and determine how data is selected and cross-classified for use by DISPLAY or other VPLX steps.

- Chapter 4 describes how the input section can specify the reading of data, both from the primary input file and through secondary linked files. In a given application, a useful programming strategy is to read in all of the variables that might be useful with a fixed set of commands, and then to modify the recode, classification, and output sections to conduct multiple analyses of the data.

- In a sense, the statements of the replication method specification section constitute the key topic of Volume II. Although this section requires at most a few statements, the selection of an appropriate replication method requires some degree of statistical training. Volume II is designed to provide first an overview and then a more detailed description of the available replication options and considerations in choosing among them. Statements illustrating common selections will appear throughout Volume I.

  On the other hand, once a replication method has been selected and appropriate statements provided in the command file, then this choice often may remain fixed for all analyses of the survey data. For example, if replicate weights are included with a public use data set from the Census Bureau, then a simple prescription for the replication specification section may be given to handle all VPLX applications for this file. Thus, in many cases the roles of analyst and statistician may be separated, with the key statistical input into the analysis reflected in the replication method specification section.

1.3 General Features of the New VPLX Syntax

The following observations about the new syntax in the CREATE step of the previous examples are true generally. Some do not apply to the old syntax, which is illustrated in the DISPLAY steps.

- Statements are delineated by ending semicolons. Statements may extend over multiple lines.

- Blank lines may be included to increase the readability of the code. Indentation of code may be used freely. It may be helpful to indent continuation lines of a statement or conditional statements following "if," for example.

- Comments following "!" may be added without changing the interpretation of the statement to the left. In particular, "!" does not have the effect of an ending semicolon. Extended comments may appear between "/*" and "*/".

- CREATE and other steps begin with an initial statement starting in position 1. Other statements in the step are indented, that is, they must not use position 1. (Metastatements, to be introduced later, also begin in position 1.)

- Except for character strings and file names under UNIX, VPLX statements are generally case insensitive. For example, the documentation often makes reference to a "CREATE statement" even though the corresponding example may include "create," because case makes no difference in the interpretation.

C The length of records on the command file must not exceed 80 characters. VPLX will check that this limit has not been exceeded by scanning for any additional non-blank characters in positions 81-100, to detect most accidental transgressions of this rule.

1.4 The CREATE Statement

A statement with CREATE starting in position 1 signals the beginning of a CREATE step. The statement must also specify two files. One file is an existing character input file containing numeric data and the other is the output VPLX file. The program will terminate in error if the input file does not exist, but the output file may be either a new or existing file. If the output file exists, it will be overwritten.

As is the case generally with file specifications for VPLX steps, the output file must be different from the input file.

The syntax is:

```
CREATE   IN = fname1  OUT = fname2 ;
```

The length of each file name is limited to 80 characters. The form of *fname1* and *fname2* depends on the host system. In many environments, either specification may be the full name or shortened name (depending on default directories, *etc.*) of the file.

Section xx details the specification of files for VPLX, including system-dependent features. A few examples are given here, however, as a guide.

In WIN 95/NT environment, the CREATE statement might take the form:

```
create  in = c:\survey\datafile.dat
        out = c:\survey\datafile.vpl ;
```

or, if c:\survey is the current directory, simply

```
create  in = datafile.dat   out = datafile.vpl;
```

As long as full names are specified, however, it is not necessary for the two files to be in the same directory or DOS drive. For example, the input file may reside on a CD-ROM drive, `e:`,while the output file must be directed to a drive for which the user has write access, such as a standard hard disk, `c:`. The .vpl extension on the file name is recommended as a way of identifying VPLX files, but this convention is optional.

Naming of files under UNIX is similar, although not identical, to DOS. (For example, UNIX employs "/" instead of "\" in naming subdirectories.)

```
create  in = /data5/datafile.dat
        out = /tmp/datafile.vpl;
```

UNIX distinguishes between lower and upper case in naming files, so that file names must be spelled in the correct case, unlike the PC environment. In most other respects, however, VPLX is case-insensitive.

In a VAX VMS environment,

```
create  in  = user$:[user_name.survey]datafile.dat
          out = temp$:[user_name]datafile.vpl ;
```

illustrates the naming of files. If the VMS file name includes a ";" to denote version, then the file name must be included in single quotes to prevent the interpretation of the semicolon as the end of the CREATE statement.

```
create  in  = 'user$:[user_name.survey]datafile.dat;3'
          out = temp$:[user_name]datafile.vpl ;
```

Default directories or shortened names may be used.

1.5 Recommended File Extensions.

Note that in DOS, UNIX, and VMS, among other systems, the last part of a file name is often called the extension and used to distinguish types of files. VPLX does not enforce any particular system of file extensions. On the other hand, the author finds that consistent use of a convention is quite helpful. The author's own preferences are:

.DAT for a character (e.g., ASCII) data set.

.VPL for VPLX files output from VPLX.

.CRD for command files (*2*).

.VSK for files containing VPLX code (similar to .CRD) with unresolved substitutions requiring the SET feature of VPLX (*3*). These files are analogous to macros in some other languages. In a production system, a .CRD file may contain SET statements to define the substitution strings for the particular run, and then reference one or more .VSK files through INCLUDE.

.LIS for output print files, "listings," from VPLX. These files could be sent to a printer, although the author typically reads them with an editor and commits few to paper.

1.6 Variable Names.

VPLX variable names must begin with a letter, and contain a combination of 1-12 letters, underscores, "_", and digits. A few variable names are not allowed: AS, BLOCK, BY, CLASS, FOR, IF, INTO, KEY, MEAN, MEANS, MINUS, N, OPTION, OPTIONS, PERCENT, PERCENTS, PERCENT1, PERCENT2, PLUS, PROPORTION, PROPORTIONS, PROPORTION1, TOTAL, TOTALS, and TOTAL1, since these are elements of the syntax (*4*). Longer names, such as MEAN_INCOME, that imbed any of these terms are acceptable. (Except for the allowed length of 12 instead of 8, these rules follow those for SAS variable names.) Note that the minus sign ("-") and other special symbols are not allowed as part of a variable name.

VPLX treats upper and lower case spellings as equivalent; for example, `MEAN_INCOME`, `mean_income`, and even `Mean_income` all refer to the same variable.

Some variable names, such as WEIGHT, CLUSTER, STRATUM, can and should be used with reserved meanings assigned by VPLX. For example, WEIGHT identifies a variable normally to be used as a weight for the observations. Nonetheless, in some cases statements in VPLX allow these reserved meanings to be overridden.

Variable names REPW, REPW0, REPW1, REPW2, ..., or REPF, REPF0, REPF1, REPF2, ..., have an entirely reserved meaning, and can only be used to represent replicate weights or factors, respectively. Volume II explains these further.

Some other variable names have reserved meanings in specific contexts. For example, when a HADAMARD statement appears in a CREATE or REWEIGHT step, variable names ROW1, ROW2, ... and COEF1, COEF2, ... *etc.*, refer to variables to be used to create either replicate factors or weights. Outside this context, however, the names ROW1, ..., COEF1, *etc.*, do not have a reserved use.

1.7 A More Extensive Example.

Most examples in this documentation employ very small data sets. Although small examples are easy to follow and can often be replicated by hand, they may not successfully convey the flavor of larger VPLX applications. For this purpose, an extract from the 1987 Full Panel Microdata Research File from the Survey of Income and Program Participation (SIPP) (*5*) is also provided. The extract contains only a few of the variables on the CD-ROM but all of the cases.

An example from the extract is included here to illustrate how the standard order applies to a more complex application. The reader may find the example partially but not entirely self-explanatory. Subsequent chapters will return to points illustrated by this example and other applications based on the same extract.

```
! i1-3.crd

   ... scratch files set up

create  in = sipp87x.dat
        out = sipp1.vpl ;

    input rot  su_id  pp_entry  pp_num  pp_intvw1 - pp_intvw4
        pp_mis1 - pp_mis15  pnlwgt fnlwgt87 hsc  strat
        sex  race  ethnicity  rrp1 - rrp15
        age1 - age15  ms1 - ms15  higrade1 - higrade4
        grd_cmpl1 - grd_cmpl4  in_af1 - in_af4  tenure1 - tenure15
        pp_inc1 - pp_inc4  pp_earn1 - pp_earn4  ff_inc1 - ff_inc4
        ff_povd1 - ff_povd4  esr1 - esr15 / options nprint =1 /

    format (f1.0,f9.0,       !  rot  1   su_id  2-10
            f2.0,f3.0,       !  pp_entry  11-12   pp_num  13-15
            4f1.0,           !  pp_intvw1-4   16-19
            15f1.0,          !  pp_mis1-15    20-34
            2f12.7,          !  pnlwgt 35-46   fnlwgt87  47-58
            f1.0,f2.0,       !  hsc  59   strat  60-61
            2f1.0,f2.0,      !  sex  62   race  63   ethnicity  64-65
            15f1.0,15f2.0,   !  rrp1-15       66-80   age1-15     81-110
            15f1.0,4f2.0,    !  ms1-15      111-125   higrade1-4 126-133
            4f1.0,4f1.0,     !  grd_cmpl1-4 134-137   in_af1-4   138-141
            15f1.0,4f8.0,    !  tenure1-15  142-156   pp_inc1-4  157-188
            4f7.0,4f8.0,     !  pp_earn1-4  189-216   ff_inc1-4  217-248
            4f5.0,15f1.0)  ; !  ff_povd1-4  249-268   esr1-15    269-283

    if hsc == 1 then;             !  Specification for modified half
       coef1 = -.5 ;              !  sample.  72 pseudo strata are
    else;                         !  identified on the file.  Each
       coef1 = .5 ;               !  pseudo stratum is divided into
    end if;                       !  half samples with hsc= 1 or 2.
    hadamard  72;
    coefficients 72 * .055555555556 ;
    row1 = strat;
    weight pnlwgt;

    /* The 4 rotation groups of SIPP are interviewed in different
       months.  Rotation group 2 was first interviewed in Feb 1987
       to cover Oct 86 - Jan 87.  Rotations 3, 4, and 1 were interviewed
       in Mar, Apr, and May, respectively.  The following code
       creates monthly variables covering the 1987 calendar year. */

    if rot == 2 then;
      {c_pp_mis1 - c_pp_mis12} = {pp_mis4 - pp_mis15};
      {c_rrp1 - c_rrp12} = {rrp4 - rrp15};
      {c_age1 - c_age12} = {age4 - age15};
      {c_ms1 - c_ms12} = {ms4 - ms15};
      {c_tenure1 - c_tenure12} = {tenure4 - tenure15};
      {c_esr1 - c_esr12} = {esr4 - esr15};
```

```
else if rot == 3 then;
  {c_pp_mis1 - c_pp_mis12} = {pp_mis3 - pp_mis14};
  {c_rrp1 - c_rrp12} = {rrp3 - rrp14};
  {c_age1 - c_age12} = {age3 - age14};
  {c_ms1 - c_ms12} = {ms3 - ms14};
  {c_tenure1 - c_tenure12} = {tenure3 - tenure14};
  {c_esr1 - c_esr12} = {esr3 - esr14};

else if rot == 4 then;
  {c_pp_mis1 - c_pp_mis12} = {pp_mis2 - pp_mis13};
  {c_rrp1 - c_rrp12} = {rrp2 - rrp13};
  {c_age1 - c_age12} = {age2 - age13};
  {c_ms1 - c_ms12} = {ms2 - ms13};
  {c_tenure1 - c_tenure12} = {tenure2 - tenure13};
  {c_esr1 - c_esr12} = {esr2 - esr13};

else;
  {c_pp_mis1 - c_pp_mis12} = {pp_mis1 - pp_mis12};
  {c_rrp1 - c_rrp12} = {rrp1 - rrp12};
  {c_age1 - c_age12} = {age1 - age12};
  {c_ms1 - c_ms12} = {ms1 - ms12};
  {c_tenure1 - c_tenure12} = {tenure1 - tenure12};
  {c_esr1 - c_esr12} = {esr1 - esr12};
end if;

/* The following code adjusts highest grade to completed
   grade.  Highest grade is measured at each interview
   rather than each month. */

if pp_intvw1 .in. {1,2} .and. age1 >= 15 then;
   if grd_cmpl1 == 1 then;                        ! completed grade
     hicmpgrd1 = higrade1 ;
   else if grd_cmpl1 == 2 then;                   ! grade not complete
      if higrade1 == 0 .or. higrade1 == 21 .or.   ! do not adjust
         higrade1 == 25 then;                     ! 21 (some college)
        hicmpgrd1 = higrade1;                     ! 25 (some grad sch)
      else;
        hicmpgrd1 = higrade1 - 1;                 ! adjust other grades
      end if;
   end if;
end if;
if pp_intvw2 .in. {1,2} .and. age5 >= 15 then;
   if grd_cmpl2 == 1 then;
      hicmpgrd2 = higrade2 ;
   else if grd_cmpl2 == 2 then;
      if higrade2 == 0 .or. higrade2 == 21 .or.
         higrade2 == 25 then;
        hicmpgrd2 = higrade2;
      else;
        hicmpgrd2 = higrade2 - 1;
      end if;
   end if;
end if;

...  Similar recodes for hicmpgrd3 and hicmpgrd4

class sex (1/2) 'Male' 'Female';
class race (1/2/3/4) 'White' 'Black' 'Amer. Indians'
```

```
                    'Asian and Pacific Islanders';
class ethnicity (14-20/res) 'Hispanic origin' 'Non-Hispanic';
class c_tenure1 - c_tenure12 (2/res) 'Renter' 'Owner or other';
cat  c_esr1 - c_esr12 (1/2/3/4/5/6/7/8) 'Worked all weeks'
       'Job,miss 1+ wk,no layoff'
       'Job, some time on layoff'
       'Part job, no layoff/look'
       'Part job, w layoff/look'
       'No job, all look/layoff'
       'No jb, some look/layoff'
       'No job, no look/layoff';
label c_esr1 'Jan. 1987 Labor Force Status'
      c_esr2 'Feb. 1987 Labor Force Status'
      c_esr3 'Mar. 1987 Labor Force Status'
      c_esr4 'Apr. 1987 Labor Force Status'
      c_esr5 'May 1987 Labor Force Status'
      c_esr6 'Jun. 1987 Labor Force Status'
      c_esr7 'Jul. 1987 Labor Force Status'
      c_esr8 'Aug. 1987 Labor Force Status'
      c_esr9 'Sep. 1987 Labor Force Status'
      c_esr10 'Oct. 1987 Labor Force Status'
      c_esr11 'Nov. 1987 Labor Force Status'
      c_esr12 'Dec. 1987 Labor Force Status';
cat hicmpgrd1 - hicmpgrd4 (0-11/12/21-23/24/25-26)
             'Less than HS' 'HS graduate' 'Some college'
             'College grad' 'Some post-grad';
label hicmpgrd1 'Completed ed., intvw 1'
      hicmpgrd2 'Completed ed., intvw 2'
      hicmpgrd3 'Completed ed., intvw 3'
      hicmpgrd4 'Completed ed., intvw 4' ;

block hicmpgrd1 / select if pp_intvw1 .in. {1}
               / select if age1 .in. {15- high};
block hicmpgrd2 / select if pp_intvw2 .in. {1}
               / select if age5 .in. {15- high};
block hicmpgrd3 / select if pp_intvw3 .in. {1}
               / select if age9 .in. {15- high};
block hicmpgrd4 / select if pp_intvw4 .in. {1}
               / select if age13 .in. {15- high};
block c_esr1 / class sex * race * ethnicity
      / select if c_pp_mis1 .in. {1}
      / select if c_age1 .in. {15-high};
block c_esr2 / class sex * race * ethnicity
      / select if c_pp_mis2 .in. {1}
      / select if c_age2 .in. {15-high};

... Similar block statements for months 3-12

Generalized replication assumed

Size of block   1  =             6

Size of block   2  =             6

Size of block   3  =             6

Size of block   4  =             6

Size of block   5  =           144
```

```
    ... Similar results for blocks 6-15

    Total size of tally matrix =   1752

**** End of CREATE specification/beginning of execution

Observation     1 from unit 12
     rot                 4.0000          su_id         1074034.0000
     pp_entry           11.0000          pp_num            101.0000
     pp_intvw1           1.0000          pp_intvw2           1.0000
     pp_intvw3           1.0000          pp_intvw4           1.0000
     pp_mis1             1.0000          pp_mis2             1.0000
     pp_mis3             1.0000          pp_mis4             1.0000
     pp_mis5             1.0000          pp_mis6             1.0000
     pp_mis7             1.0000          pp_mis8             1.0000
     pp_mis9             1.0000          pp_mis10            1.0000
     pp_mis11            1.0000          pp_mis12            1.0000
     pp_mis13            1.0000          pp_mis14            1.0000
     pp_mis15            1.0000          pnlwgt              7.7677
     fnlwgt87            7.2960          hsc                 2.0000
     strat              40.0000          sex                 2.0000
     race                1.0000          ethnicity           4.0000
     rrp1                2.0000          rrp2                2.0000
     rrp3                2.0000          rrp4                2.0000
     rrp5                2.0000          rrp6                2.0000
     rrp7                2.0000          rrp8                2.0000
     rrp9                2.0000          rrp10               2.0000
     rrp11               2.0000          rrp12               2.0000
     rrp13               2.0000          rrp14               2.0000
     rrp15               2.0000          age1               48.0000
     age2               48.0000          age3               48.0000
     age4               49.0000          age5               49.0000
     age6               49.0000          age7               49.0000
     age8               49.0000          age9               49.0000
     age10              49.0000          age11              49.0000
     age12              49.0000          age13              49.0000
     age14              49.0000          age15              49.0000
     ms1                 4.0000          ms2                 4.0000
     ms3                 4.0000          ms4                 4.0000
     ms5                 4.0000          ms6                 4.0000
     ms7                 4.0000          ms8                 4.0000
     ms9                 4.0000          ms10                4.0000
     ms11                4.0000          ms12                4.0000
     ms13                4.0000          ms14                4.0000
     ms15                4.0000          higrade1           12.0000
     higrade2           12.0000          higrade3           12.0000
     higrade4           12.0000          grd_cmpl1           1.0000
     grd_cmpl2           1.0000          grd_cmpl3           1.0000
     grd_cmpl4           1.0000          in_af1               .0000
     in_af2               .0000          in_af3               .0000
     in_af4               .0000          tenure1             2.0000
     tenure2             2.0000          tenure3             2.0000
     tenure4             2.0000          tenure5             2.0000
     tenure6             2.0000          tenure7             2.0000
     tenure8             2.0000          tenure9             2.0000
     tenure10            2.0000          tenure11            2.0000
     tenure12            2.0000          tenure13            2.0000
     tenure14            2.0000          tenure15            2.0000
     pp_inc1          1542.0000          pp_inc2          1926.0000
```

```
     pp_inc3          1542.0000          pp_inc4          1543.0000
     pp_earn1         1536.0000          pp_earn2         1920.0000
     pp_earn3         1536.0000          pp_earn4         1536.0000
     ff_inc1          1542.0000          ff_inc2          1926.0000
     ff_inc3          1542.0000          ff_inc4          1543.0000
     ff_povd1         5748.0000          ff_povd2         5784.0000
     ff_povd3         5808.0000          ff_povd4         5832.0000
     esr1                1.0000          esr2                1.0000
     esr3                1.0000          esr4                1.0000
     esr5                1.0000          esr6                1.0000
     esr7                1.0000          esr8                1.0000
     esr9                1.0000          esr10               1.0000
     esr11               1.0000          esr12               1.0000
     esr13               1.0000          esr14               1.0000
     esr15               1.0000
(Printing discontinued on unit 12)

    End of primary input file after obs #   26441



display

list  c_esr1 - c_esr12 / class total /
      c_esr1 / class sex race(2) ethnicity (1)

list hicmpgrd1 - hicmpgrd4



                                                  Estimate      Standard error

Jan. 1987 Labor Force St:  PERCENTS
   Worked all weeks                              57.7070              .4861
   Job,miss 1+ wk,no layoff                       1.5299              .0992
   Job, some time on layoff                        .3364              .0510
   Part job, no layoff/look                        .6520              .0526
   Part job, w layoff/look                         .6700              .0754
   No job, all look/layoff                        3.6330              .1612
   No jb, some look/layoff                         .6528              .0682
   No job, no look/layoff                        34.8188              .4665

Feb. 1987 Labor Force St:  PERCENTS
   Worked all weeks                              58.4732              .4936
   Job,miss 1+ wk,no layoff                       1.0650              .0888
   Job, some time on layoff                        .2486              .0414
   Part job, no layoff/look                        .5029              .0531
   Part job, w layoff/look                         .6693              .0718
   No job, all look/layoff                        3.6213              .1376
   No jb, some look/layoff                         .5448              .0668
   No job, no look/layoff                        34.8748              .4751

... Similar displays for March - December
```

sex : Male

	Estimate	Standard error
Jan. 1987 Labor Force St: PERCENTS		
Worked all weeks	66.5366	.6005
Job,miss 1+ wk,no layoff	1.7254	.1529
Job, some time on layoff	.4904	.0847
Part job, no layoff/look	.4917	.0707
Part job, w layoff/look	.7242	.1131
No job, all look/layoff	4.5256	.2390
No jb, some look/layoff	.6726	.1118
No job, no look/layoff	24.8335	.5542

sex : Female

	Estimate	Standard error
Jan. 1987 Labor Force St: PERCENTS		
Worked all weeks	49.6324	.5676
Job,miss 1+ wk,no layoff	1.3512	.1223
Job, some time on layoff	.1956	.0480
Part job, no layoff/look	.7986	.0886
Part job, w layoff/look	.6204	.0869
No job, all look/layoff	2.8168	.1979
No jb, some look/layoff	.6347	.0902
No job, no look/layoff	43.9503	.5829

race : Black

	Estimate	Standard error
Jan. 1987 Labor Force St: PERCENTS		
Worked all weeks	50.1697	1.5068
Job,miss 1+ wk,no layoff	.9930	.3056
Job, some time on layoff	.5106	.2154
Part job, no layoff/look	.4521	.1878
Part job, w layoff/look	1.2504	.3467
No job, all look/layoff	7.3006	.8611
No jb, some look/layoff	1.7594	.3341
No job, no look/layoff	37.5642	1.4057

ethnicity : Hispanic origin

	Estimate	Standard error
Jan. 1987 Labor Force St: PERCENTS		
Worked all weeks	56.7422	1.8899
Job,miss 1+ wk,no layoff	1.3044	.3620
Job, some time on layoff	.3239	.1707
Part job, no layoff/look	.8227	.2668
Part job, w layoff/look	.8150	.2858
No job, all look/layoff	4.9681	.8588
No jb, some look/layoff	.7583	.2957
No job, no look/layoff	34.2654	1.7395

	Estimate	Standard error
Completed ed., intvw 1 : PERCENTS		
Less than HS	25.8180	.5240
HS graduate	34.0098	.4617
Some college	22.2100	.3438
College grad	9.1430	.2843
Some post-grad	8.8193	.2695

```
Completed ed., intvw 2  :  PERCENTS
   Less than HS                                  25.1351              .5695
   HS graduate                                   33.4993              .4755
   Some college                                  23.2307              .3632
   College grad                                   9.2134              .3068
   Some post-grad                                 8.9214              .2628

Completed ed., intvw 3  :  PERCENTS
   Less than HS                                  24.4345              .5754
   HS graduate                                   33.5663              .4791
   Some college                                  23.6196              .3904
   College grad                                   9.1476              .2832
   Some post-grad                                 9.2320              .2636

Completed ed., intvw 4  :  PERCENTS
   Less than HS                                  24.1432              .5842
   HS graduate                                   33.3471              .5145
   Some college                                  23.6670              .3856
   College grad                                   9.2989              .2893
   Some post-grad                                 9.5438              .2817
```

Exhibit 1.5 Example print file, i1-3.lis, from the 1987 SIPP panel file. Some blank lines and repetitive sections (noted in italics) have been omitted. The input section reads all of the variables from an extract file. The application uses variables from the public use file to create modified half-sample replicates. An extensive recode section defines a series of variables for the 1987 calendar year and defines a new variable for completed education. The classification and output sections define relevant universes and cross-classifiers for the completed education and labor force variables. The DISPLAY selectively illustrates only some of the potential results that could be calculated from the VPLX file.

The 1987 panel of the SIPP is a longitudinal sample based seven interviews conducted 4 months apart. The file contains both monthly data, such as labor force status, and variables measured at each interview, such as educational attainment. Each of 4 rotation groups had a different starting month. Only 15 of the months and 4 interviews are included for the selected variables. The example illustrates how the labor force data may be sorted out into longitudinal data for the calendar year 1987.

Full versions of i1-3.crd and i1-3.lis are available in the example files. Example i1-4.lis shows the modifications in the input section to obtain the same results directly from the CD-ROM file instead of the extract file.

NOTES

1. The next revision of the VPLX metadata may be based on XML (Extensible Markup Language) so that VPLX may readily communicate this information to other systems. At this date, XML is a proposed standard of the World Wide Web Consortium.

2. .CRD is an abbreviation of CARD. In the early 1970's, input to programs was often through 80-column computer punch cards. Rubber bands, receptacles for cards, *etc*., were standard equipment of the era. The author finds the abbreviation a useful mnemonic, but the user is free to choose another convention.

3. VSK may be remembered as a *VPLX skeleton*. The skeleton shows the outlines of the procedure but requires external set statements to flesh out the details.

4. In most cases, VPLX will notice attempts to use these reserved names and display an error message.

5. Survey of Income and Program Participation (SIPP) 1987 Full Panel Microdata Research File on CD-ROM, prepared by the Data User Services Division, Bureau of the Census, Washington, DC, 1993.