A STATISTICAL CROSSTABULATION INTERFACE FOR INGRES: A BEGINNING

Linda Lehnen
William A. Gates

CDE Working Paper 87-2

A STATISTICAL CROSSTABULATION INTERFACE FOR INGRES: A BEGINNING

Linda Lehnen and William A. Gates

A Statistical Crosstabulation Interface for INGRES: A Beginning

Linda M. Lehnen and William A. Gates
Center for Demography and Ecology,
University of Wisconsin-Madison

Statistical analysis in Demography routinely involves hundreds of thousands of observations. It is typical in the Social Sciences that statistical analysis involves tens of thousands of observations.

The use of statistical packages to construct the proper set of observations is difficult, time consuming and expensive. We have found QUEL to be a much more effective tool for this task. The statistical packages are much more efficient at the analytical stage.

Frequently the first statistical step, once the set of observations has been created, is to produce crosstabulations for preliminary discovery of the relationships among the variables in the sample or population under study. It is desirable to minimize the resources required to accomplish this step because often it leads to a revision in the observation set or to the construction of a new unit of analysis and then a new observation set. This led us to develop an interface between INGRES and a crosstabulation package (local to our environment) called XTAB. Now a researcher can create the set of observations and efficiently perform analysis while capturing results all within the database environment provided by INGRES.

This paper reports on how the interface is used, its performance relative to statistical packages, and its performance relative to INGRES aggregate operators.

This research was carried out (in part) using facilities of the Center for Demography and Ecology at the University of Wisconsin-Madison, which receives core support for Population Research from the National Institute for Child Health and Human Development (HD-5876). Presented at the INGRES Users Conference, Minneapolis, MN, April 1986.

A Statistical Crosstabulation Interface for Ingres: A Beginning

Introduction

The main motivation for developing an interface between a cross-tabulation package and INGRES was the rather routine observation that in our environment most researchers used INGRES to deal with the complexities of forming the proper unit of analysis and then copied that set of observations from INGRES into a standard VAX VMS file. Usually the first analytical step is to perform countless cross-tabulations on this file. Frequently a conceptual problem or error of logic is discovered and the researcher then returns to the database to discover the misunderstanding, correct it and extract once again. Statistical analysis of many datasets routine to demographic studies involve hundreds of thousands or even millions of observations, e.g., 1980 U.S. Census of Population. We also observed that many "canned" statistical packages were now performing all calculations from their own "system files" and in double precision, thus adding significantly to resource use and the length of time required to complete the process. Saving the effort and resources associated with prematurely unloading data and beginning to process it with the statistical packages seemed like a worthy, high-return investment.

Although the scope of statistical packages has been broadened considerably to cope with complex data structures they remain principally focused on applying a statistical procedure to a rectangular set of data. Few statistical packages provide for even read-only user concurrency. A few statistical packages support some sort of random access to files. This simply does not compare with the efficiencies of a database system, such as INGRES, or database query language, such as QUEL. The ability to examine data at the individual case (row) level or linkage between different levels of observation (household and person) for particular cases is of paramount importance to discover problems of logic or conceptualization. The ability to then concisely, with a few steps and perhaps only a few hundred keystrokes, create an entirely new set of observations from many different relations is extraordinarily powerful.

Development Strategy

The objective was to demonstrate the feasibility of generally interfacing statistical procedures with INGRES. A second related but independent objective was to provide a better command language interface with a local cross-tabulation package called XTAB. This language would then be extended to serve as the interface with INGRES. We limited the the scope of the initial project to obtaining crosstabulation tables from a single INGRES table with a where clause for conditions pertaining only to that table. Two months elapsed time were allowed for design, implementation, testing and benchmarking. A good deal of the time was actually spent on developing and writing a proposal

on what would be done and then during implementation adhering to the philosophy to record the "better ideas" and save them for future enhancements. As some revisions were made, our "planning document" was also revised. This resulted in six revisions to our original plan.

We chose XTAB as the statistical package to be interfaced with INGRES for the following reasons:

1. It is the most efficient statistical package in use for crosstabulations on our system.
2. We have access to the XTAB source code.
3. Feasible to interface interface XTAB with INGRES using EQUEL/FORTRAN since XTAB is written in Fortran.
4. Its original language interface was entirely card-oriented and thus quite difficult for terminal users to setup and run.

Description of the New XTAB

It has already been mentioned that a related objective of creating an XTAB-INGRES interface was to create a new language front-end for XTAB. XTAB has had a long history being revised many times to run on many different computer systems since it was first written in the early 1960s. Interestingly the fixed format card-oriented interface had never been revised. This has served as a great impediment to its use even though it was many times faster than any alternative. Along the way an external description of the format of the data has been replaced by a Fortran driver. Although this has even made using XTAB more cumbersome, it has the advantage of making it possible to add basic transformations to the step of generating the cross-tabulations.

The following set of XTAB commands produce frequency tables and column and row percentage tables for sex by age by race for those respondents who have been in the labor force, worked from 50-52 weeks in the year and earned between $1 and $75000. Tables containing means for SALA and HRS are also produced.

```
RUN MARG=YES ZERO=NO
TITLE ADULT PUS801000 TEST XTAB
NAME EMPL 3
BOUNDARY (1,2,INLF)
NAME WKWK 4
BOUNDARY (50,52,FLYR)
NAME WGSL 6
BOUNDARY (1,75000)
NAME SEX 7
```

```
INTERVAL LOWER=0,UPPER=1,STEP=1 MALE FEML
NAME AGE 8
INTERVAL LOWER=15,UPPER=99,STEP=5
NAME RACE 9
BOUNDARY (01,01,WHIT) (02,02,BLAK) (03,13,OTHR)
TABLE OPTION = ALL
DIMENSION SEX AGE RACE EMPL WKWK WGSL
TABLE_TITLE AGE X SEX X RACE — EMPLOYED 50-52 WEEKS
   WAGES > 0
ASSOCVAR (SALA,6) (HRS,5)
```

The NAME control card is used to define a variable. The second field of the name card indicates the location of the variable in XTAB_ARRAY, which is used in the user's driver program to hold data passed to the cross-tabulation software. In the example, the user's Fortran program would contain the following assignment statements:

```
XTAB_ARRAY(3) = EMPL
XTAB_ARRAY(4) = WKWK
XTAB_ARRAY(6) = WGSL
XTAB_ARRAY(7) = SEX
XTAB_ARRAY(8) = AGE
XTAB_ARRAY(9) = RACE
```

Each NAME command must have either an INTERVAL or a BOUNDARY command associated with it. If the values are to be broken down into categories of unequal length, then the BOUNDARY command is utilized. Otherwise the INTERVAL command is specified. The BOUNDARY and INTERVAL commands can also be used to select a subsample of the population. Only records whose values are in the specified range will be included in the sample. EMPL will be 1 or 2 if the person is in the labor force. The BOUNDARY card following NAME EMPL 3 will limit the sample for the table to those who are in the labor force.

The TABLE command specifies that a table should be created. The option ALL requests that percentages are generated for row and column totals in addition to the frequency table. A DIMENSION command must be associated with a table command to indicate which variables to use in the table. The ASSOCVAR command is optional. It should be included if dependent variables are to be included. In the example, a table will be generated that contains cell means for SALA and another table will be generated that contains cell means for HRS.

The XTAB-INGRES Interface

In order to implement the interface between INGRES and XTAB, a new language, INGXTB, was designed. This language consists of the XTAB commands, modified slightly, plus four new commands. The syntax of the language is as follows:

EDIT (don't execute, just check syntax)

DATABASE databasename

TABLE tablename

ATTRIBUTES list (list contains names of all attributes that will be used
continue if necessary in XTAB tables.)

WHERE condition (optional)
Continue the condition on as many lines as necessary.

RUN MARG=yes/no ZERO=yes/no (default yes for MARG, no for ZERO)

TITLE title (optional)

NAME name attribute_name (attribute_name indicates the INGRES attribute name listed on the ATTRIBUTE command.)

BOUNDARY (lower1,upper1,name1) (lower2,upper2,name2) ...
(continue more bounds on successive lines if necessary.)

INTERVAL LOWER=lower UPPER=upper STEP=step name1 name2
name3 name4 ...
(continue more names on successive lines if necessary)

repeat previous NAME, BOUNDARY or INTERVAL commands for each variable.
There must be either a BOUNDARY or INTERVAL command, but not both.

All NAME, BOUNDARY, and INTERVAL commands must precede any
TABLE commands.

TABLE OPTION=keyword WEIGHT=attribute_name,name
where keyword = FREQ (print raw table, this is default)
ROW (print table and % of row total)
COL (print table and % of column total)
ALL (print table, % row and column total)

DIMENSION col_var_name row_var_name page1_var_name ... page13_var_name
(continue names on successive lines if necessary)

TABLE_TITLE (optional)

ASSOCVAR (name1,attribute_name1) (name2,attribute_name2) ... (optional)
(continue on successive lines if necessary.)

The table lines must be in the order described, i.e., TABLE line followed by DIMENSION optionally followed by TABLE_TITLE and/or ASSOCVAR. Repeat TABLE, DIMENSION, TABLE_TITLE, and ASSOCVAR for each table.

The following set of INGXTB commands produce the same tables that were generated by the previous XTAB example. However, now the data is obtained from the table L1ADULT80 in the PUS801000 database.

```
DATABASE PUS801000
TABLE L1ADULT80
ATTRIBUTES LABOR WEEKSW79 HOURS79 INCOME1 SEX
   AGE RACE
RUN MARG=YES ZERO=NO
TITLE ADULT PUS801000 TEST XTAB-INGRES
NAME EMPL LABOR
BOUNDARY (1,2,INLF)
NAME WKWK WEEKSW79
BOUNDARY (50,52,FLYR)
NAME WGSL INCOME1
BOUNDARY (1,75000)
NAME SEX SEX
INTERVAL LOWER=0,UPPER=1,STEP=1 MALE FEML
NAME AGE AGE
INTERVAL LOWER=15,UPPER=99,STEP=5
NAME RACE RACE
BOUNDARY (01,01,WHIT) (02,02,BLAK) (03,13,OTHR)
TABLE OPTION = ALL
DIMENSION SEX AGE RACE EMPL WKWK WGSL
TABLE_TITLE AGE X SEX X RACE - EMPLOYED 50-52 WEEKS
   WAGES > 0
ASSOCVAR (SALA,INCOME1) (HRS,HOURS79)
```

Using INGXTB

INGXTB generates an EQUEL/FORTRAN program, TEMPINGB.QBF while the four new commands (DATABASE, TABLE, ATTRIBUTES, and WHERE) are being processed. This program replaces the XTAB driver program and the FORTRAN program that was set up by the user. In addition, the "XTAB" commands are preprocessed before being passed to XTAB. Note that in INGXTB, the NAME and ASSOCVAR commands now use attribute_name rather than array location. The process of describing variable locations for XTAB that was accomplished by the user's FORTRAN program is now accomplished by TEMPINGB.QBF. INGXTB manages the details of choosing array locations and passing that information on to XTAB'S cross-tabulation code. As the ATTRIBUTE command is processed, the attribute names are stored in the array ATTRIB. A RETRIEVE statement is generated for the list of attributes and placed in TEMPINGB.QBF. This RETRIEVE statement stores the attribute in the location in XTAB_ARRAY that corresponds to its location in ATTRIB. When INGXTB processes the NAME and ASSOCVAR commands, the attribute_name is replaced with its location in ATTRIB before the command is passed to XTAB. XTAB variable names are restricted to four characters. Thus it is not possible to simply replace the XTAB variable name with the INGRES attribute name without actually modifying the XTAB code. XTAB is written in FORTRAN IV and poorly documented. Thus we chose to write a preprocessor rather than modify the code.

Following is the EQUEL/FORTRAN program generated by INGXTB when the previous set of commands were processed:

```
            program TEMPINGQBF
            implicit logical (a-z)
            integer handle
!           declare ingres variables
##          declare
##          real*4 XTAB_ARRAY(32767)
##          integer cpu_time,dirio
!           zero counts and xtab tables
!
            handle = 0
            call XTAB(-1,XTAB_ARRAY)
!
!           get ingres database
##          ingres 'PUS801000'
##          range of r is L1ADULT80
##          retrieve(cpu_time=_cpu_ms,dirio=_dio_cnt)
            print *,'before retrieve cpu =',cpu_time,'ms',
           1 ' dio = ',dirio
```

```
            if(.not. lib$init_timer(handle)) print *,'init err'
!           retrieve the records
##          retrieve(
##          XTAB_ARRAY( 1)=r.LABOR
##          ,XTAB_ARRAY( 2)=r.WEEKSW79
##          ,XTAB_ARRAY( 3)=r.HOURS79
##          ,XTAB_ARRAY( 4)=r.INCOME1
##          ,XTAB_ARRAY( 5)=r.SEX
##          ,XTAB_ARRAY( 6)=r.AGE
##          ,XTAB_ARRAY( 7)=r.RACE
##          )
!
!           set up retrieve loop
!
##          {
            call XTAB(0,XTAB_ARRAY)
 ##         }
!
##          retrieve(cpu_time=_cpu_ms,dirio=_dio_cnt)
            print *,'after retrieve cpu =',cpu_time, 'ms',
           1 ' dio = ',dirio
            if (.not. lib$show_timer(handle)) print *,'err timer'
!           print the tables
!
            call XTAB(1,XTAB_ARRAY)
            if (.not. lib$show_timer(handle)) print *,'err timer'
##          exit
            end
```

In order to use INGXTB, the user sets up the control cards in a file and enters @INGXTB control_file The command file INGXTB contains commands to assign temporary files, compile, link, execute and delete temporary files.

Benchmarking and Comparisons

Runs were set up to complete the same task using the interface INGXTB, SPSSX, and XTAB. The task was to obtain six tables for selected populations from a sample of 1980 Census Public Use data. The first four tables generated means for income and hours worked per week for different combinations of the variables sex, age, race and education. The sample was restricted to those who had been in the labor force, worked from 50-52 weeks in the year and earned between $1 and $75000. The fifth table contained frequencies for number of children by age at first marriage for those women who were 35-99 years old. The sixth table contained frequencies for number of children by age at

first marriage by race for those women who were 35-99 years old. To obtain accurate benchmarks, the tests were run on 10, 20 and 30% samples.

The following tables show the results of these runs (VAX 11/870 VMS 4.2 with 8 megabytes):

10% Sample
17472 rows, 5884 rows after selection (tables 1–4),
4758 rows after selection (tables 5–6)

PACKAGE	CPU secs	DIO	PAGE FAULTS
INGXTB	256	835	3399
XTAB	64	559	1448
SPSSX	297	679	5291

20% Sample
34944 rows, 11856 rows after selection (tables 1–4),
9438 rows after selection (tables 5–6)

PACKAGE	CPU secs	DIO	PAGE FAULTS
INGXTB	509	841	3133
XTAB	112	660	1350
SPSSX	538	967	5276

30% Sample
52417 rows, 17907 rows after selection (tables 1–4),
14093 rows after selection (tables 5–6)

PACKAGE	CPU secs	DIO	PAGE FAULTS
INGXTB	742	866	3219
XTAB	160	754	1382
SPSSX	798	1251	5510

As the tables show, XTAB performed much better than either INGXTB or SPSSX. We ignored the resources required to copy out from INGRES in order to run XTAB and the time required to copy in to INGRES in order to run INGXTB. These two should average out in the long run. INGXTB did perform slightly better than SPSSX.

For purposes of rough procedural comparison, we also solved a subset of this task using aggregate operators within INGRES.

In order to use INGRES aggregate operators to do a crosstabulation, the following steps must be performed:

1. Use RETRIEVE INTO with WHERE to select desired rows.
2. Use REPLACE to recode values to desired ranges.
3. Use RETRIEVE with AVG and COUNT to get counts and cell means.

The INGRES code to solve a task similar to the previous XTAB and INGXTB examples follows:

```
range of s is l1adult80
retrieve into subpus80 (s.all) where (s.labor = 1 or s.labor =2) and
(s.weeksw79 >= 50 and s.weeksw79 <=52) and
(s.income1 >=1 and s.income1 <=75000)
\g
range of p is subpus80
replace p(age = 1) where p.age >= 15 and p.age <= 19
replace p(age = 2) where p.age >= 20 and p.age <= 24
replace p(age = 3) where p.age >= 25 and p.age <= 29
replace p(age = 4) where p.age >= 30 and p.age <= 34
replace p(age = 5) where p.age >= 35 and p.age <= 39
replace p(age = 6) where p.age >= 40 and p.age <= 44
replace p(age = 7) where p.age >= 45 and p.age <= 49
replace p(age = 8) where p.age >= 50 and p.age <= 54
replace p(age = 9) where p.age >= 55 and p.age <= 59
replace p(age = 10) where p.age >= 60 and p.age <= 64
replace p(age = 11) where p.age >= 65 and p.age <= 69
replace p(age = 12) where p.age >= 70 and p.age <= 74
replace p(age = 13) where p.age >= 75 and p.age <= 79
replace p(age = 14) where p.age >= 80 and p.age <= 84
replace p(age = 15) where p.age >= 85 and p.age <= 89
replace p(age = 16) where p.age >= 90 and p.age <= 94
replace p(age = 17) where p.age >= 95 and p.age <= 99
replace p(race=3) where p.race >=3 and p.race <= 13
\g
retrieve (p.sex,p.age,p.race,cnt=count(p.sex by p.sex,p.age,p.race),
avginc=avg(p.income1 by p.sex,p.age,p.race),
avghrs = avg(p.hours79 by p.sex,p.age,p.race))
\g
```

```
range of p is subpus80
\g
retrieve (p.sex,cntsex=count(p.sex by p.sex))
retrieve (p.age,cntage=count(p.age by p.age))
retrieve (p.race,cntrace=count(p.race by p.race))
retrieve (avgsex=avg(p.sex),avgage=avg(p.age),avgrace=avg(p.race))
\g
```

This method does not generate the column or row frequencies or standard deviations that are computed by statistical packages.

An INGRES job using AGGREGATES was also set up to generate six tables similar to those produced by the benchmark runs. The INGRES run on the 10% sample required 1046 seconds of CPU time. This same task only required 256 seconds using INGXTB. The INGRES run did not generate as much information since column and row frequencies and standard deviations were not computed. In addition the INGRES run required 729 blocks to store INGRES tables generated by RETRIEVE's when selecting desired sample. INGXTB does not require these temporary files since selection can be controlled by BOUNDARY and INTERVAL commands. The INGRES solution required 31 REPLACE commands to recode values that were recoded in XTAB by 6 INTERVAL and BOUNDARY commands.

Summary

As was previously mentioned, the process of developing this interface was experimental. We feel the experiment was successful. We met our time constraints, have a working, not ideal interface, and demonstrated the feasibility of an interface. Before implementing any enhancements, we plan to investigate some of the following possibilities:

1. Build on other EQUEL capabilities to allow more conditional crosstabulation based on other database tables, etc.
2. An entirely new language with new features utilizing other RTI products to provide screen management
3. Enhance/update XTAB algorithms
4. Other statistical procedure/package interfaces
5. Incorporate statistical results as relations in the database

Mailing Address:

Center for Demography and Ecology
University of Wisconsin
1180 Observatory Drive
Madison, Wisconsin 53706-1393
U.S.A.