

Indexed Datasets 101

John Charles Gober

U.S. Department of Commerce, Bureau of the Census
Longitudinal, Expenditures and Victimization Surveys Branch

Synopsis

This paper is a shortened version of the paper presented at SESUG97 in Nashville under the title Understanding Indexed Datasets and Using Direct Access Queries. It will briefly introduce programmers to the bare minimum code needed to create and implement direct access processing through indexing. After reviewing this paper, the beginning to average programmer should have enough knowledge of the indexing process to be able to accomplish simple direct access queries which, under the proper circumstances, will save considerable processing time over using the customary merge routine.

When to use Direct Accessing

Retrieval by direct access is most efficient when a relatively small number of records from a transaction dataset need to be integrated or merged with an exponentially larger master dataset. For example, the Bureau of the Census conducts numerous non-decennial surveys for many of its sister agencies consisting of upward to several million participants. For quality assurance purposes a small sample of these participants are re-interview and their responses compared to the originals.

By only accessing the records on the master file that are known to be on the transaction file, processing time can be significantly reduced, sometimes in the area of over 90 percent. The principal is the same as going to the grocery store and going directly to the item on the aisle instead of starting at aisle one and working your way up.

Creating the index

There are two methods of creating indexes. Method one is by including the necessary code

as one of the options when declaring a dataset as shown in Figure One.

```
LIBNAME REGION10 "/fieldata/";
LIBNAME CENSUS99 "/decennial/";

DATA CENSUS99.VA22308
  (INDEX=(HEADHLD/UNIQUE
          ZIPPLUS4));
SET REGION10.VA
  (WHERE=(ZIP=22308));
...more program code;
RUN;
```

Figure One

SAS®, when detecting this style of code, will process all programming statements as is done under normal processing, create all output datasets as specified in the DATA statement, and then, as a final process create an extra dataset containing the unique index for HEADHLD and non unique index for ZIPPLUS4.

This index creation process may or may not be beneficial to the use for the following reason. When using this method of index creation, one must remember that the creation process will add extra CPU time to the dataset sometimes doubling the processing time. If there are numerous time critical jobs that will be executed after this process which do not require direct accessing it might be wise to use the second method of creating an index as in the sample below and place the code after all non index processing has been done.

```
LIBNAME CENSUS99 "/decennial/";

PROC DATASETS LIBRARY=CENSUS99;
  MODIFY VA22308,
  INDEX CREATE HEADHLD/UNIQUE;
  INDEX CREATE ZIPPLUS4;
RUN;
```

Figure Two

This second method of index creation has several possible advantages over the first. It can be executed independently of the data step thus freeing up precious CPU time, and the PROC DATASTEP can be placed in a final administrative program at the end of a cycle of jobs.

Using the Index

If you have used an index in the back of a book then you have used the same technique that the SAS System uses to directly access records. *At this point it must be mentioned that indexed datasets are actually two paired datasets, one containing the actual data, the other containing the indices and physical record location.* Figure Three contains the basic code for the direct access of records in an indexed dataset.

```
DATA WORK.ADDRESS
  SET CENSUS99.TRANS;
  SET CENSUS99.VA22308
    KEY=HEADHLD / UNIQUE;
  IF (_IORC_ NE %SYSRC(_DSENOM))
    THEN OUTPUT;
    ELSE DELETE;
RUN;
```

Figure Three

Both datasets CENSUS99.TRANS and CENSUS99.VA22308 contain the variable HEADHLD. When SAS begins execution of the dataset it first reads a record from the TRANS file (which should have respectively fewer number of records than the indexed file). Based on the value of the variable HEADHLD, SAS then reads the index file of VA22308. If a matching record is found in the index then the exact physical location is determined of the data file of VA22308 and the record is directly read bypassing all records which may be ahead of it.

For ease of programming, SAS has supplied a macro which resides in the SASAUTOS library. This macro, called %sysrc, contains definitions of many of the system return codes which SAS uses for processing. One of these return codes, '1230015', has been assigned to the variable _DSENOM. By comparing this variable with the value of the system code returned by the index read one can determine if the operation was successful.

Figure Four contains the basic code for accessing multiple records from a file when the index is not unique as in the case of the variable ZIPPLUS4. The code is a little more complex but uses the same basic principals of the code demonstrated in Figure Three. Notice the use of another automatic return code variable _SOK.

```
DATA BOGUS.DUPADDRS;
  SET CENSUS99.TRANZIP;
  NEWZIP=ZIPPLUS4;
  DO UNTIL
    (_IORC_=%SYSRC(_DSENOM));
    SET CENSUS99.HEADHLD
      KEY=ZIPPLUS4;
    SELECT (_IORC_);
    WHEN (%SYSRC(_SOK))
      DO;
        IF ZIPPLUS4 = NEWZIP
          THEN OUTPUT;
        END;
        WHEN (%SYSRC(_DSENOM))
          DO;
            _ERROR_ = 0;
          END;
        END;
  RUN;
```

Figure 4

Conclusion

Hopefully this paper has presented enough code to allow programmers of all experience levels to start using indexes. Remember, the code presented here is to be considered starter code. With a little thought and ingenuity it should be able to be modified to meet any scenario the programmer will come across. The author will be more than happy to try to answer any questions that may arise.

Trademarks

SAS is the registered trademark of SAS Institute Inc. in the United States and other countries. @ represents USA registration.

Author and Contact Information

John Charles Gober
Bureau of the Census
FB-3 Room 3737
4700 Silver Hill Road
Suitland MD 20746
john.c.gober@ccmail.census.gov