CHAPTER

5

Browsing and Updating CA-DATACOM/DB Data

Introduction

The SAS/ACCESS interface to CA-DATACOM/DB enables you to browse and update your CA-DATACOM/DB data directly from a SAS session or program. This chapter shows you how to use SAS procedures to browse and update CA-DATACOM/DB data described by SAS/ACCESS view descriptors.

Most of the examples in this chapter use the view descriptor VLIB.USACUST that you created in Chapter 3, “Defining SAS/ACCESS Descriptor Files,” on page 15. See Appendix 3, “Data and Descriptors for the Examples,” on page 125 for definitions of the view descriptors referenced in this chapter. This appendix also contains the CA-DATACOM/DB tables and SAS data files used in this book.

Refer to Chapter 2, “CA-DATACOM/DB Essentials,” on page 7 and Appendix 1, “Information for the Database Administrator,” on page 105 for more information on retrieval processing, update processing, and locks.

Browsing and Updating with the SAS/FSP Procedures

If your site has SAS/FSP software as well as SAS/ACCESS software, you can browse and update CA-DATACOM/DB data described by a view descriptor from within a SAS program.

You have a choice of three SAS/FSP procedures: FSBROWSE, FSEDIT, and FSVIEW. The FSBROWSE and FSEDIT procedures show you one data record at a time, while the FSVIEW procedure displays multiple records in a tabular format similar to the PRINT procedure. PROC FSVIEW enables you to browse or update CA-DATACOM/DB data, depending on which option you choose. You cannot use the FSBROWSE, FSEDIT, or FSVIEW procedures on an access descriptor.

To scroll through the data, use the FORWARD and BACKWARD commands. To end your browse or edit session, issue the END command.

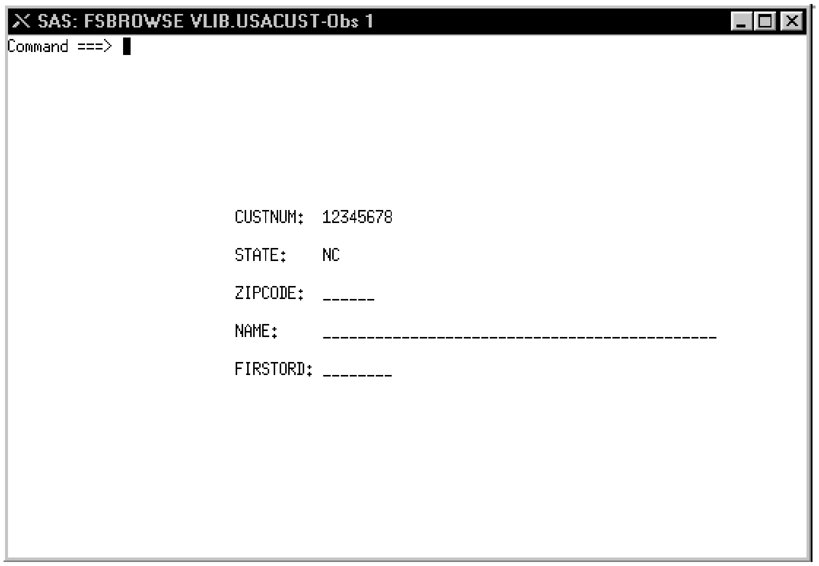
Using the FSBROWSE Procedure

The FSBROWSE procedure enables you to look at CA-DATACOM/DB data but not to change it. To use PROC FSBROWSE, submit the following SAS statements:

```
proc fsbrowse data=vlib.usacust;
run;
```

The FSBROWSE procedure retrieves one record at a time from a CA-DATACOM/DB table. Display 5.1 on page 46 shows the first record of the USA customers' data described by the VLIB.USACUST view descriptor. To browse each record, use the FORWARD and BACKWARD commands.

Display 5.1 FSBROWSE Window



SAS: FSBROWSE VLIB.USACUST-Obs 1

```
Command ===>

CUSTNUM:  12345678

STATE:    NC

ZIPCODE:  ______

NAME:     ____________________________________________

FIRSTORD: ________
```

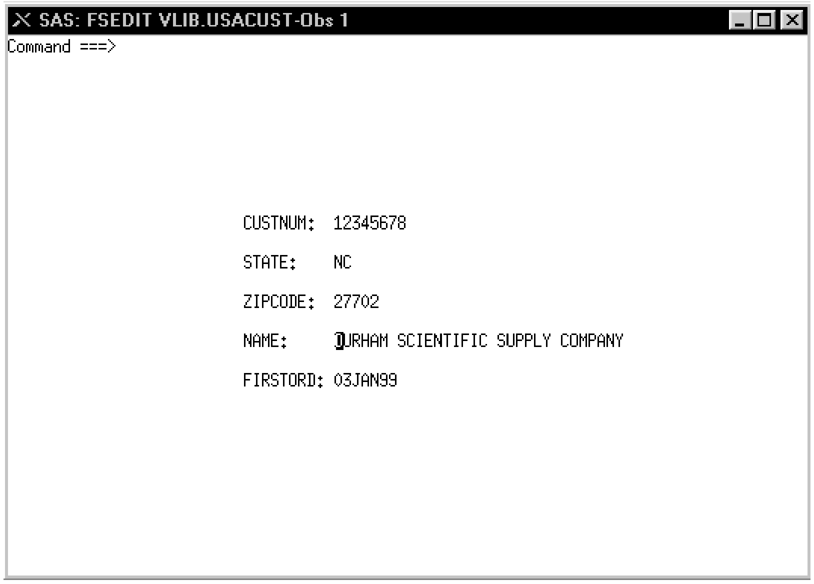
Using the FSEDIT Procedure

The FSEDIT procedure enables you to update CA-DATACOM/DB data described by a view descriptor. For example, in Display 5.1 on page 46 the ZIPCODE, NAME, and FIRSTORD values are missing in the first record. You can add values to these fields with the FSEDIT procedure.

To use PROC FSEDIT, submit the following SAS statements:

```
proc fsedit data=vlib.usacust;
run;
```

Display 5.2 FSEDIT Window



```
Command ===>

CUSTNUM:  12345678

STATE:    NC

ZIPCODE:  27702

NAME:     DURHAM SCIENTIFIC SUPPLY COMPANY

FIRSTORD: 03JAN99
```

The FSEDIT procedure also retrieves one record at a time. To edit a record, scroll to it, and type in the new data after the appropriate label. For example, enter the information about the **DURHAM SCIENTIFIC SUPPLY COMPANY**, as shown in Display 5.2 on page 47. To end your editing session, issue the END command.

Note: To cancel an edit, you must do so before you scroll. The CANCEL command does not cancel your editing changes; it redisplays the record as it was before you began to edit it. Once you scroll, the changes are committed. △

Note: The data are presented in order by the Default Key value (usually the Native Key) unless the view descriptor contains a SORT clause. △

Using the FSVIEW Procedure

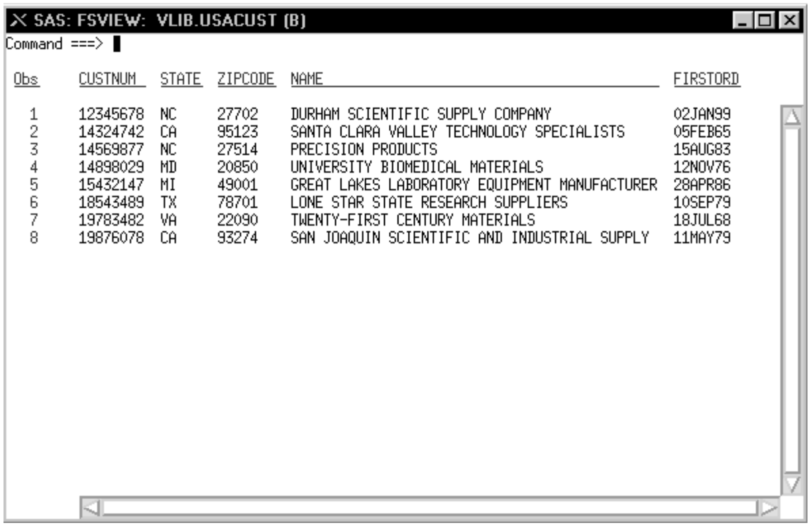
Depending on how you invoke the FSVIEW procedure, you can either browse or update data using a view descriptor.

Using the FSVIEW Procedure to Browse CA-DATACOM/DB Data

To browse CA-DATACOM/DB data, submit the PROC FSVIEW statement as follows:

```
proc fsview data=vlib.usacust;
run;
```

Display 5.3 FSVIEW Window



```
Command ===>

Obs      CUSTNUM   STATE  ZIPCODE  NAME                                       FIRSTORD

  1      12345678  NC     27702    DURHAM SCIENTIFIC SUPPLY COMPANY           02JAN99
  2      14324742  CA     95123    SANTA CLARA VALLEY TECHNOLOGY SPECIALISTS  05FEB65
  3      14569877  NC     27514    PRECISION PRODUCTS                         15AUG83
  4      14898029  MD     20850    UNIVERSITY BIOMEDICAL MATERIALS            12NOV76
  5      15432147  MI     49001    GREAT LAKES LABORATORY EQUIPMENT MANUFACTURER 28APR86
  6      18543489  TX     78701    LONE STAR STATE RESEARCH SUPPLIERS         10SEP79
  7      19783482  VA     22090    TWENTY-FIRST CENTURY MATERIALS             18JUL68
  8      19876078  CA     93274    SAN JOAQUIN SCIENTIFIC AND INDUSTRIAL SUPPLY  11MAY79
```

PROC FSVIEW displays the data in a listing format instead of one observation at a time, as shown in Display 5.3 on page 48. Browse mode is the default for the FSVIEW procedure. Notice that a (B) for browse follows the view descriptor's name and that the values in the first record reflect the changes made using the FSEDIT procedure in the previous example.

To see the rest of the table's data, scroll the display to the right several times by issuing the RIGHT command on the command line or by using the function key assigned to this command.

Note: The data are presented in order by the Default Key value (usually the Native Key) unless the view descriptor contains a SORT clause. If the view descriptor contains a WHERE clause but no SORT clause, the order is unpredictable. △

Using the FSVIEW Procedure to Update CA-DATACOM/DB Data

To edit CA-DATACOM/DB data with PROC FSVIEW, submit the FSVIEW statement as follows:

```
proc fsview data=vlib.usacust modify;
run;
```

The word "EDIT" can be used instead of MODIFY. The display will be the same as Display 5.3 on page 48 except that an (E) for edit will be displayed.

Note: Any update in the FSVIEW window is final. The CANCEL command in the FSVIEW window does *not* cancel your changes, whether or not you have scrolled. △

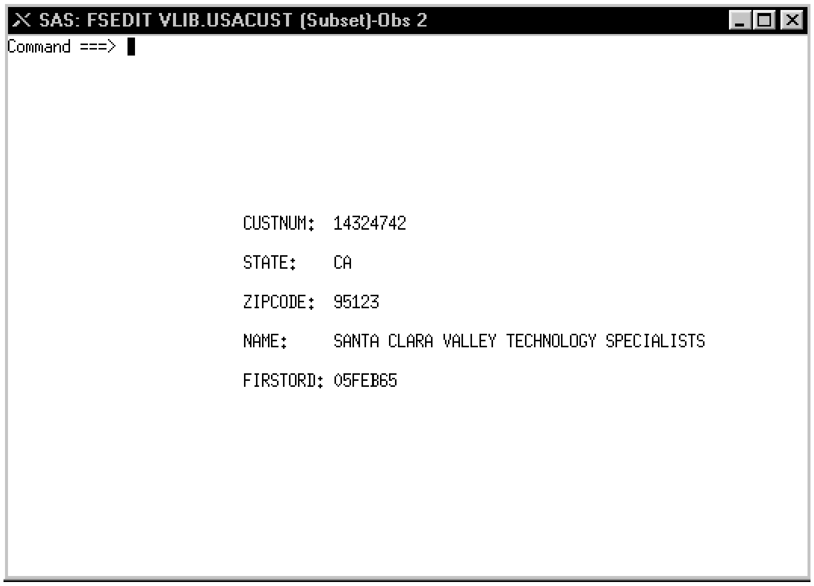
Specifying a WHERE Clause While Browsing or Editing

You can specify a WHERE statement to subset CA-DATACOM/DB data when you invoke the SAS/FSP procedures. You can also use a WHERE command to do the same thing after you have invoked one of the SAS/FSP procedures.

In the following example, a WHERE statement is used to retrieve only customers from California. Display 5.4 on page 49 shows the FSEDIT window after the statements have been submitted.

```
proc fsedit data=vlib.usacust;
   where state='CA';
run;
```

Display 5.4 FSEDIT Window After SAS WHERE Statement



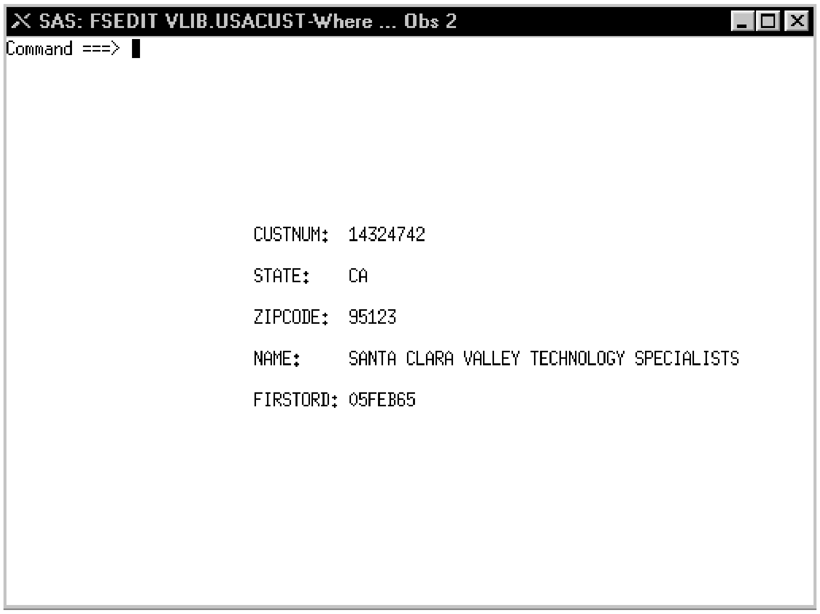
```
Command ===>

                    CUSTNUM:  14324742

                    STATE:    CA

                    ZIPCODE:  95123

                    NAME:     SANTA CLARA VALLEY TECHNOLOGY SPECIALISTS

                    FIRSTORD: 05FEB65
```

Only two records with a STATE value of CA are retrieved for editing. Note that the word (Subset) appears after VLIB.USACUST in the window title to remind you that the data retrieved are a subset of the data described by the view descriptor. You can then edit each record by typing over the information you want to modify. Issue the END command to end your editing session. If you want to cancel changes to a record, you can issue the CANCEL command before you scroll. Once you scroll though, the change is committed.

You can also use a SAS WHERE command to display a subset of your data. A WHERE command is a SAS WHERE expression that is entered on the command line. Display 5.5 on page 50 shows how the FSEDIT display appears when the subset is generated within the procedure with the following WHERE command:

```
where state='CA'
```

Display 5.5 FSEDIT Window After SAS Window After SAS WHERE Command



```
Command ===>

CUSTNUM:  14324742

STATE:    CA

ZIPCODE:  95123

NAME:     SANTA CLARA VALLEY TECHNOLOGY SPECIALISTS

FIRSTORD: 05FEB65
```

Only the two records with a STATE value of CA are retrieved for editing. Where... appears after VLIB.USACUST in the window title to remind you that the data retrieved are a subset of the data described by the view descriptor. You can then edit each record, as described earlier.

Although these examples have shown a WHERE clause with the FSEDIT procedure, you can also retrieve a subset of the data when using the FSBROWSE and FSVIEW procedures. For more information on the SAS WHERE statement, see the *SAS Language Reference: Dictionary*. For more information on the SAS WHERE command within the SAS/FSP procedures, refer to the *SAS/FSP Software: Usage and Reference*.

Inserting and Deleting Data Records with the SAS/FSP Procedures

Inserting and deleting records with the SAS/FSP procedures is different for view descriptors than for SAS data files.

You can use the FSEDIT and FSVIEW procedures to insert records into a CA-DATACOM/DB table on which a view descriptor is based. Insertion of new records also depends on the attributes assigned to the Master Key and whether the Master Key is included in your view descriptor. For example, if the DUPE-MASTER-KEY attribute is set to N (no), values for the Master Key cannot be duplicated. You will receive an error message if you try to insert a record having a Master Key value that duplicates an existing value. Therefore, be sure to define your view descriptors carefully if you intend to use them to insert records into a CA-DATACOM/DB table.

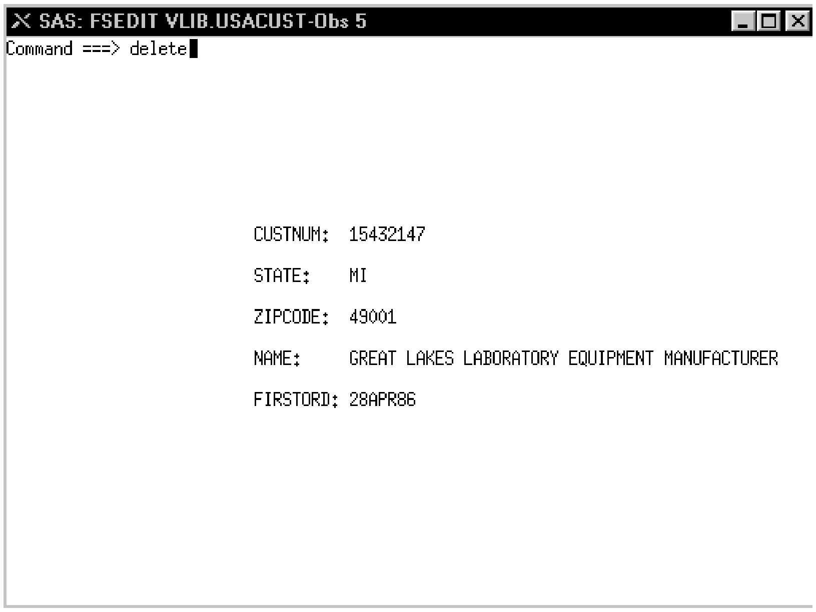
Refer to Appendix 1, “Information for the Database Administrator,” on page 105 for details on inserting data. Refer to *SAS/FSP Software: Usage and Reference* for information on how to use insertion commands such as ADD and DUP in the FSEDIT procedure and AUTOADD and DUP in the FSVIEW procedure. However, note that with the SAS/ACCESS interface to CA-DATACOM/DB, a duplicated record is inserted immediately after the original record rather than being added at the end of the CA-DATACOM/DB table.

When you use the DELETE command with a view descriptor that describes CA-DATACOM/DB data, the current record is removed permanently from the

CA-DATACOM/DB table. Also, the DELETE command works differently in the FSVIEW procedure than it does in the FSEDIT procedure. Refer to *SAS/FSP Software: Usage and Reference* for more information on this command.

The following example illustrates using the DELETE command in the FSEDIT procedure. Scroll forward to the record to be deleted and enter the DELETE command on the command line, as shown in Display 5.6 on page 51.

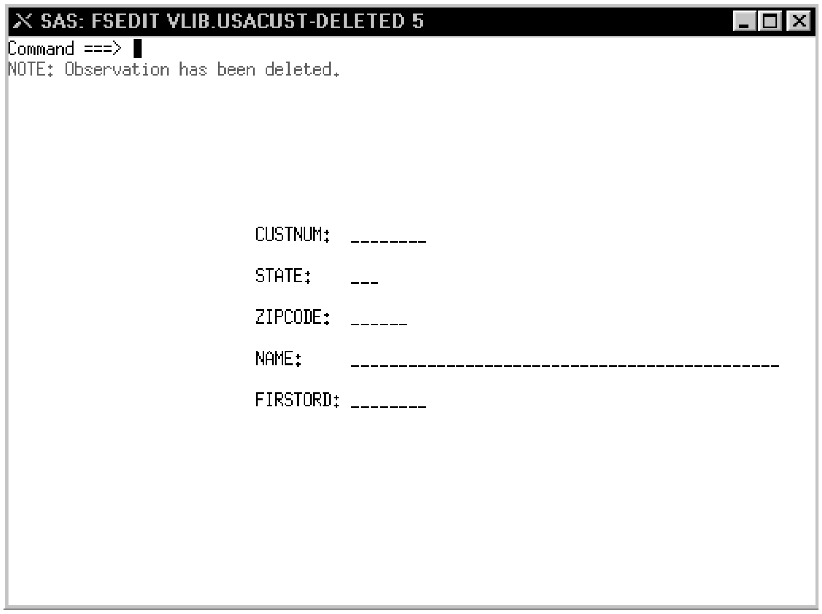
Display 5.6 Deleting a CA-DATACOM/DB Record



```
Command ===> delete

                    CUSTNUM:  15432147

                    STATE:    MI

                    ZIPCODE:  49001

                    NAME:     GREAT LAKES LABORATORY EQUIPMENT MANUFACTURER

                    FIRSTORD: 28APR86
```

The DELETE command deletes the record and displays a message to that affect, as shown in Display 5.7 on page 51.

Display 5.7 The Deleted Row



```
Command ===>
NOTE: Observation has been deleted.

                    CUSTNUM:  ________

                    STATE:    ___

                    ZIPCODE:  ______

                    NAME:     ____________________________________________

                    FIRSTORD: ________
```

For more information on using the SAS/FSP procedures, see the *SAS/FSP Software: Usage and Reference.*

Browsing and Updating with the SQL Procedure

The SAS System SQL procedure also enables you to retrieve and update CA-DATACOM/DB data. You can retrieve and browse the data by specifying a view descriptor in a PROC SQL SELECT statement.

To update the data, you can specify view descriptors in the PROC SQL INSERT, DELETE, and UPDATE statements. Here is a summary of these PROC SQL statements:

DELETE
deletes records from a CA-DATACOM/DB table.

INSERT
inserts records into a CA-DATACOM/DB table.

SELECT
retrieves and displays data from CA-DATACOM/DB tables. A SELECT statement is usually referred to as a query, because it queries the tables for information.

UPDATE
updates records in a CA-DATACOM/DB table.

When using the SQL procedure in interactive line mode, note that the data are displayed in the SAS OUTPUT window. The procedure displays output data automatically without using the PRINT procedure and executes without using the RUN statement when an SQL procedure statement is executed. You can use the QUIT statement if you want to exit the SQL procedure.

CAUTION:

When you use the SQL procedure for update processing (DELETE, INSERT, and UPDATE statements), you must set the SQL procedure option, UNDO_POLICY. The SQL procedure supports backouts of group updates for those databases that support member-level locking. CA-DATACOM/DB software does not support member-level locks. The UNDO_POLICY option allows updates to be processed without backouts. For the CA-DATACOM/DB interface, you set the value of the option to NONE. For example:

```
proc sql undo_policy=none;
  update vlib.usacust
  set zipcode=27702
  where custnum='12345678';
```

If the update is processed successfully, it is applied to the database table and a warning message is issued. The message signifies that if multiple records were updated by the command and a failure occurred some time after the first record was successfully processed, then there is no way for PROC SQL to avoid a partial update.

Partial updating means that some records are updated and some are not. It does not mean that some fields in the same record are updated while other fields are not. △

The SELECT Statement

You can use the SELECT statement to browse CA-DATACOM/DB data described by a view descriptor. The query in the following example retrieves and displays all the fields and records in the CUSTOMERS table that are described by the VLIB.USACUST view descriptor. The UNDO_POLICY procedure option is included to disable member-level locking and enable updates later in the PROC SQL execution. You can exclude the

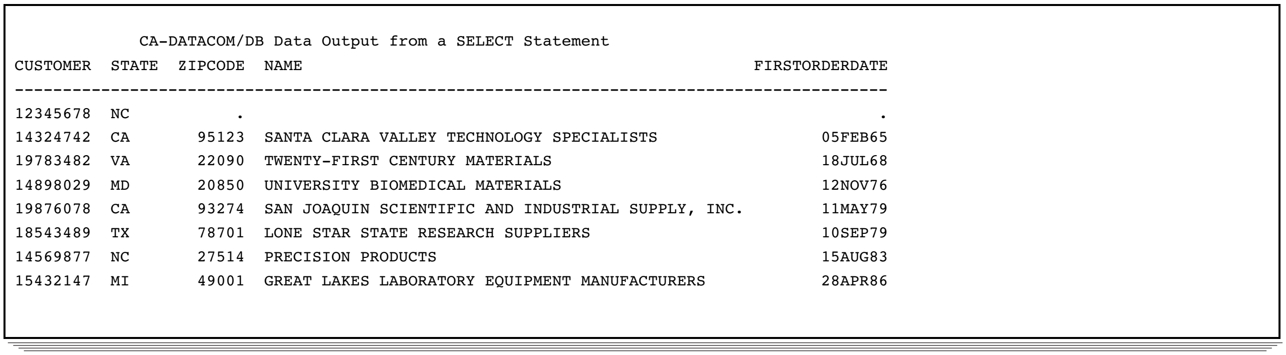
UNDO_POLICY option if you do not plan to perform updates. The LINESIZE= system option is used to reset the default output width to 120 columns.

Note: The following SQL procedure examples assume the CUSTOMERS table has not been updated by the earlier SAS/FSP examples. △

```
options linesize=120

proc sql undo_policy=none;
   title 'CA---DATACOM/DB Data Output from a SELECT Statement';
select custnum, state label='STATE', zipcode label='ZIPCODE',
       name, firstord
   from vlib.usacust;
```

Output 5.1 on page 53 displays the query's results. Notice that the SQL procedure displays the CA-DATACOM/DB field names, not the corresponding SAS column names.

Output 5.1 CA-DATACOM/DB. Data Output from a PROC SQL Query



```
             CA-DATACOM/DB Data Output from a SELECT Statement
CUSTOMER  STATE  ZIPCODE  NAME                                              FIRSTORDERDATE
------------------------------------------------------------------------------------------
12345678  NC           .                                                                 .
14324742  CA       95123  SANTA CLARA VALLEY TECHNOLOGY SPECIALISTS                05FEB65
19783482  VA       22090  TWENTY-FIRST CENTURY MATERIALS                           18JUL68
14898029  MD       20850  UNIVERSITY BIOMEDICAL MATERIALS                          12NOV76
19876078  CA       93274  SAN JOAQUIN SCIENTIFIC AND INDUSTRIAL SUPPLY, INC.       11MAY79
18543489  TX       78701  LONE STAR STATE RESEARCH SUPPLIERS                       10SEP79
14569877  NC       27514  PRECISION PRODUCTS                                       15AUG83
15432147  MI       49001  GREAT LAKES LABORATORY EQUIPMENT MANUFACTURERS           28APR86
```

You can specify a WHERE clause as part of the SELECT statement to subset the records for display. This example displays the companies that are located in North Carolina.

```
title 'CA---DATACOM/DB Data Output Subset by a WHERE Clause';
select custnum, state label='STATE', zipcode label='ZIPCODE',
       name, firstord
   from vlib.usacust
   where state='NC';
```

Notice that the PROC SQL statement is not repeated in this query. You do not need to repeat the PROC statement unless you use another SAS procedure, the DATA step, or a QUIT statement between PROC SQL statements. Output 5.2 on page 54 displays the two companies from North Carolina described by VLIB.USACUST.

Output 5.2 CA-DATACOM/DB. Data Output Subset by a WHERE Clause

```
 CA-DATACOM/DB Data Output Subset by a WHERE Clause
CUSTOMER  STATE  ZIPCODE  NAME                   FIRSTORDERDATE
---------------------------------------------------------------
12345678  NC           .                                      .
14569877  NC       27514  PRECISION PRODUCTS            15AUG83
```

The UPDATE Statement

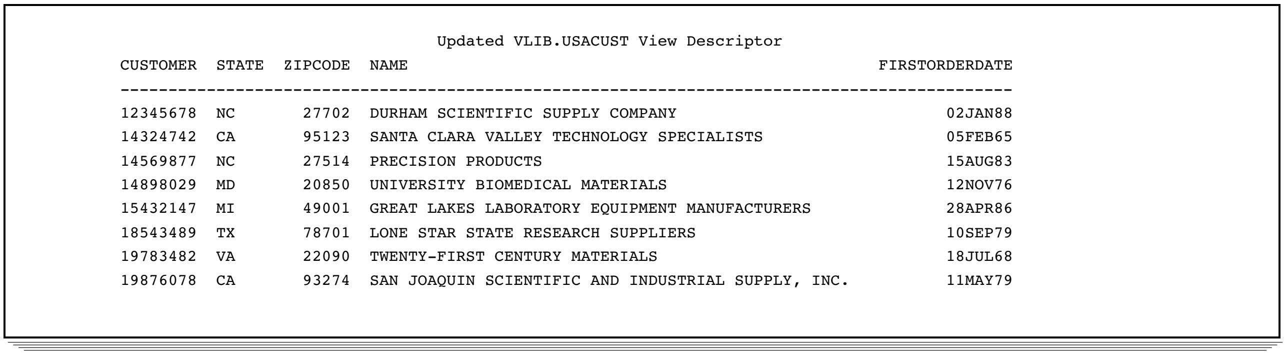
You can use the UPDATE statement to update CA-DATACOM/DB data. Remember that when you reference a view descriptor in a PROC SQL statement, you are not updating the view descriptor, but rather the CA-DATACOM/DB data described by the view descriptor.

The following UPDATE statements update the values described by the first record of VLIB.USACUST. The SELECT statement then displays the view's output. The ORDER BY clause in the SELECT statement causes the data to be presented in ascending order by the CUSTNUM field. The UNDO_POLICY option is omitted since it was specified in the original SQL request.

```
update vlib.usacust
   set zipcode=27702
   where custnum='12345678';
update vlib.usacust
   set name='DURHAM SCIENTIFIC SUPPLY COMPANY'
   where custnum='12345678';
update vlib.usacust
   set firstord='02jan88'd
   where custnum='12345678';
   title 'Updated VLIB.USACUST View Descriptor';
select custnum, state label='STATE', zipcode label='ZIPCODE', name,
   firstord from vlib.usacust
   order by custnum;
```

Output 5.3 on page 55 displays the query's results.

Output 5.3 Updated VLIB.USACUST View Descriptor



```
                                    Updated VLIB.USACUST View Descriptor
            CUSTOMER  STATE  ZIPCODE  NAME                                                  FIRSTORDERDATE
            ---------------------------------------------------------------------------------------------
            12345678  NC       27702  DURHAM SCIENTIFIC SUPPLY COMPANY                               02JAN88
            14324742  CA       95123  SANTA CLARA VALLEY TECHNOLOGY SPECIALISTS                      05FEB65
            14569877  NC       27514  PRECISION PRODUCTS                                             15AUG83
            14898029  MD       20850  UNIVERSITY BIOMEDICAL MATERIALS                                12NOV76
            15432147  MI       49001  GREAT LAKES LABORATORY EQUIPMENT MANUFACTURERS                 28APR86
            18543489  TX       78701  LONE STAR STATE RESEARCH SUPPLIERS                             10SEP79
            19783482  VA       22090  TWENTY-FIRST CENTURY MATERIALS                                 18JUL68
            19876078  CA       93274  SAN JOAQUIN SCIENTIFIC AND INDUSTRIAL SUPPLY, INC.             11MAY79
```

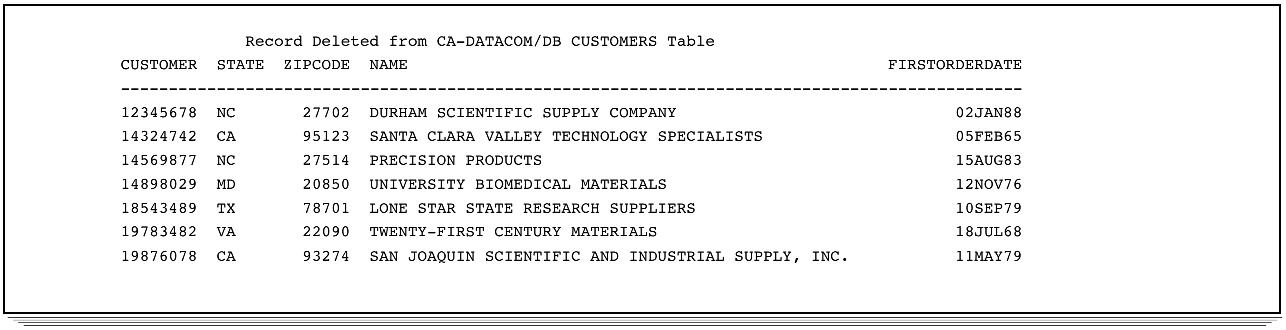
The INSERT and DELETE Statements

You can use the INSERT statement to add records to a CA-DATACOM/DB table or the DELETE statement to remove records. In the following example, the record containing the CUSTNUM value 15432147 is deleted from the table CUSTOMERS. The SELECT statement then displays the VLIB.USACUST data, ordering them again by the CUSTNUM field. Again, the UNDO_POLICY option was omitted because it was specified in the original SQL request and no intervening SAS procedure, DATA step, or QUIT statement occurred between SQL statements.

```
delete from vlib.usacust
   where custnum='15432147';
title 'Record Deleted from CA-DATACOM/DB CUSTOMERS Table';
select custnum, state label='STATE', zipcode label='ZIPCODE',
       name, firstord
   from vlib.usacust
   order by custnum;
```

Output 5.4 on page 56 displays the query's results.

Output 5.4 VLIB.USACUST Data with a Record Deleted



```
             Record Deleted from CA-DATACOM/DB CUSTOMERS Table
CUSTOMER  STATE  ZIPCODE  NAME                                                  FIRSTORDERDATE
----------------------------------------------------------------------------------------------
12345678  NC       27702  DURHAM SCIENTIFIC SUPPLY COMPANY                             02JAN88
14324742  CA       95123  SANTA CLARA VALLEY TECHNOLOGY SPECIALISTS                    05FEB65
14569877  NC       27514  PRECISION PRODUCTS                                           15AUG83
14898029  MD       20850  UNIVERSITY BIOMEDICAL MATERIALS                              12NOV76
18543489  TX       78701  LONE STAR STATE RESEARCH SUPPLIERS                           10SEP79
19783482  VA       22090  TWENTY-FIRST CENTURY MATERIALS                               18JUL68
19876078  CA       93274  SAN JOAQUIN SCIENTIFIC AND INDUSTRIAL SUPPLY, INC.           11MAY79
```

CAUTION:

Always use the WHERE clause in a DELETE statement. If you omit the WHERE clause from the DELETE statement, you will delete *all* the data in the CA-DATACOM/DB table accessed by the view descriptor. △

For more information on the SAS System SQL procedure, see the *SAS Procedures Guide.*

Appending Data with the APPEND Procedure

In releases of the SAS System earlier than Version 6, the APPEND procedure operated only on SAS data files. You can now append data described by SAS/ACCESS view descriptors and PROC SQL views to SAS data files and vice versa. You can also append data described by view descriptors to each other.

The input file and base file do not have to match column for column. If they do not match, use the FORCE option in the APPEND procedure. This will include all columns in the base file. Values for columns that are not shared by the base and input files are set to missing.

In the following example, two personnel managers have kept separate employee records. One manager has kept records in the CA-DATACOM/DB table EMPLOYEES, described by the view descriptor VLIB.DCMEMPS. The other manager has kept records in a Version 6 SAS data file, MYDATA.SASEMPS. Due to a corporate reorganization, the two sources of data must be combined so that all employee data are stored in the CA-DATACOM/DB table EMPLOYEES. The APPEND procedure can perform this task.

The data described by the view descriptor VLIB.DCMEMPS and the data in the SAS data file MYDATA.SASEMPS are printed with the following statements and displayed in Output 5.5 on page 57 and Output 5.6 on page 57.

```
proc print data=vlib.dcmemps;
   title 'Data Described by VLIB.DCMEMPS';
run;

proc print data=mydata.sasemps;
```

```
   format birthdat date7.;
   title 'Data in MYDATA.SASEMPS Data File';
run;
```

Output 5.5 Data Described by VLIB.DCMEMPS

```
                         Data Described by VLIB.DCMEMPS                          1
OBS    EMPID   BIRTHDAT   LASTNAME              FIRSTNAM          MIDDLENA

  1   119012   05JAN46    WOLF-PROVENZA         G.                ANDREA
  2   120591   12FEB46    HAMMERSTEIN           S.                RACHAEL
  3   123456         .    VARGAS                PAUL              JESUS
  4   127845   25DEC43    MEDER                 VLADIMIR          JORAN
  5   129540   31JUL60    CHOULAI               CLARA             JANE
  6   135673   21MAR61    HEMESLY               STEPHANIE         J.
  7   212916   29MAY28    WACHBERGER            MARIE-LOUISE      TERESA
  8   216382   24JUL63    PURINTON              PRUDENCE          VALENTINE
  9   234967   21DEC67    SMITH                 GILBERT           IRVINE
 10   237642   13MAR54    BATTERSBY             R.                STEPHEN
 11   239185   28AUG59    DOS REMEDIOS          LEONARD           WESLEY
 12   254896   06APR49    TAYLOR-HUNYADI        ITO               MISHIMA
 13   321783   03JUN35    GONZALES              GUILLERMO         RICARDO
 14   328140   02JUN51    MEDINA-SIDONIA        MARGARET          ROSE
 15   346917   15MAR50    SHIEKELESLAM          SHALA             Y.
 16   356134   25OCT60    DUNNETT               CHRISTINE         MARIE
 17   423286   31OCT64    MIFUNE                YUKIO             TOSHIRO
 18   456910   24SEP53    ARDIS                 RICHARD           BINGHAM
 19   456921   12MAY62    KRAUSE                KARL-HEINZ        G.
 20   457232   15OCT63    LOVELL                WILLIAM           SINCLAIR
 21   459287   15JAN34    RODRIGUES             JUAN              M.
 22   677890   24APR65    NISHIMATSU-LYNCH      CAROL             ANNE
```

Output 5.6 Data in MYDATA.SASEMPS

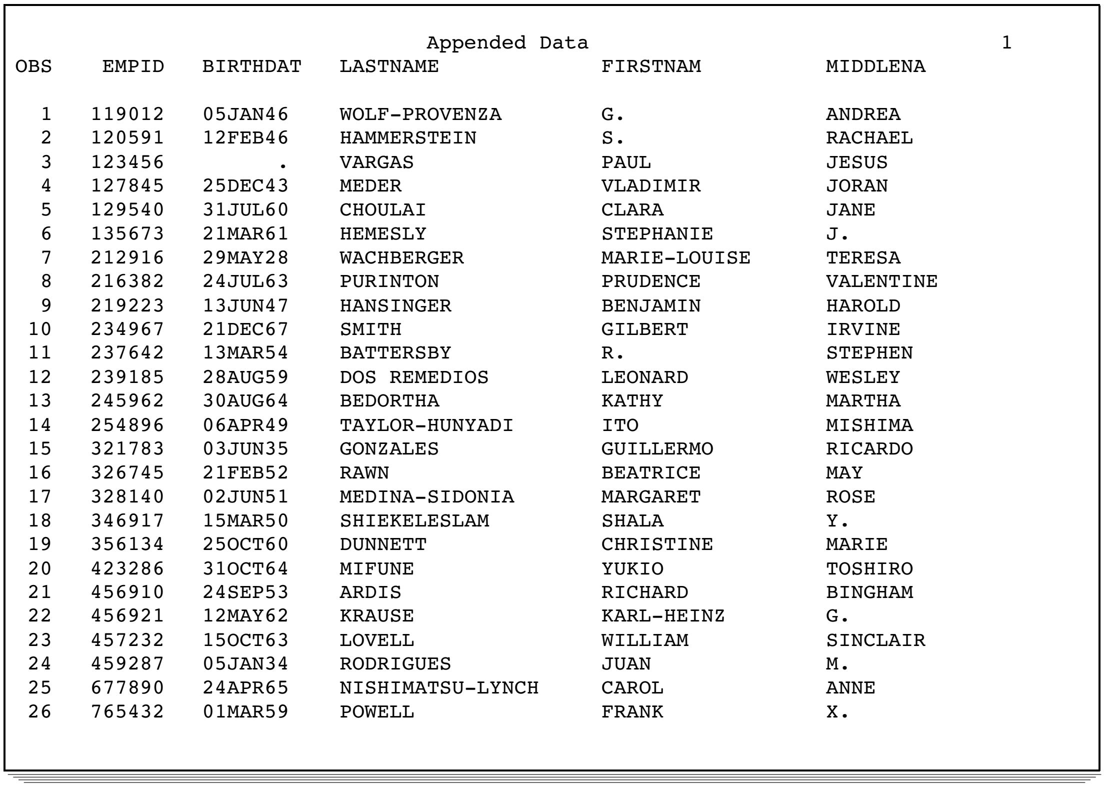
```
                        Data in MYDATA.SASEMPS Data File                         1
         OBS      EMPID    BIRTHDAT    LASTNAME     FIRSTNAM    MIDDLENA

          1      245962    30AUG64     BEDORTHA     KATHY        MARTHA
          2      765432    01MAR59     POWELL       FRANK        X.
          3      219223    13JUN47     HANSINGER    BENJAMIN     HAROLD
          4      326745    21FEB52     RAWN         BEATRICE     MAY
```

Submitting the following statements with the APPEND procedure combines data from these two sources:

```
proc append base=vlib.dcmemps data=mydata.sasemps;
run;

proc print data=vlib.dcmemps;
   title 'Appended Data';
run;
```

Output 5.7 on page 58 displays the appended data described by the view descriptor VLIB.DCMEMPS. Notice that the data were inserted in the order of Native Key values.

Output 5.7 Appended Data



```
                                  Appended Data                                    1
OBS    EMPID   BIRTHDAT   LASTNAME            FIRSTNAM          MIDDLENA

  1   119012   05JAN46    WOLF-PROVENZA       G.                ANDREA
  2   120591   12FEB46    HAMMERSTEIN         S.                RACHAEL
  3   123456         .    VARGAS              PAUL              JESUS
  4   127845   25DEC43    MEDER               VLADIMIR          JORAN
  5   129540   31JUL60    CHOULAI             CLARA             JANE
  6   135673   21MAR61    HEMESLY             STEPHANIE         J.
  7   212916   29MAY28    WACHBERGER          MARIE-LOUISE      TERESA
  8   216382   24JUL63    PURINTON            PRUDENCE          VALENTINE
  9   219223   13JUN47    HANSINGER           BENJAMIN          HAROLD
 10   234967   21DEC67    SMITH               GILBERT           IRVINE
 11   237642   13MAR54    BATTERSBY           R.                STEPHEN
 12   239185   28AUG59    DOS REMEDIOS        LEONARD           WESLEY
 13   245962   30AUG64    BEDORTHA            KATHY             MARTHA
 14   254896   06APR49    TAYLOR-HUNYADI      ITO               MISHIMA
 15   321783   03JUN35    GONZALES            GUILLERMO         RICARDO
 16   326745   21FEB52    RAWN                BEATRICE          MAY
 17   328140   02JUN51    MEDINA-SIDONIA      MARGARET          ROSE
 18   346917   15MAR50    SHIEKELESLAM        SHALA             Y.
 19   356134   25OCT60    DUNNETT             CHRISTINE         MARIE
 20   423286   31OCT64    MIFUNE              YUKIO             TOSHIRO
 21   456910   24SEP53    ARDIS               RICHARD           BINGHAM
 22   456921   12MAY62    KRAUSE              KARL-HEINZ        G.
 23   457232   15OCT63    LOVELL              WILLIAM           SINCLAIR
 24   459287   05JAN34    RODRIGUES           JUAN              M.
 25   677890   24APR65    NISHIMATSU-LYNCH    CAROL             ANNE
 26   765432   01MAR59    POWELL              FRANK             X.
```

The APPEND procedure also accepts a WHERE= data set option or a WHERE statement to retrieve a subset of data. In the following example, a subset of observations from the DATA= data set is added to the BASE= data set. The results are displayed in Output 5.8 on page 59.

```
proc append base=vlib.dcmemps data=mydata.sasemps
   (where=(lastname like 'B%' or lastname like 'H%'));
```

```
run;

proc print data=vlib.dcmemps;
   title 'Appended Data';
run;
```

Output 5.8 on page 59 displays the data when the observations appended to the BASE= data set are subset by the WHERE= data set option. In this case, the WHERE= data set option specified that only the employees with last names beginning with B or H should be added to the BASE= data set.

Output 5.8 Appended Data with a WHERE= Data Set Option

```
                                  Appended Data                                  1
OBS   EMPID    BIRTHDAT   LASTNAME             FIRSTNAM          MIDDLENA

  1  119012    05JAN46    WOLF-PROVENZA        G.                ANDREA
  2  120591    12FEB46    HAMMERSTEIN          S.                RACHAEL
  3  123456          .    VARGAS               PAUL              JESUS
  4  127845    25DEC43    MEDER                VLADIMIR          JORAN
  5  129540    31JUL60    CHOULAI              CLARA             JANE
  6  135673    21MAR61    HEMESLY              STEPHANIE         J.
  7  212916    29MAY28    WACHBERGER           MARIE-LOUISE      TERESA
  8  216382    24JUL63    PURINTON             PRUDENCE          VALENTINE
  9  219223    13JUN46    HANSINGER            BENJAMIN          HAROLD
 10  234967    21DEC67    SMITH                GILBERT           IRVINE
 11  237642    13MAR54    BATTERSBY            R.                STEPHEN
 12  239185    28AUG59    DOS REMEDIOS         LEONARD           WESLEY
 13  245962    30AUG64    BEDORTHA             KATHY             MARTHA
 14  254896    06APR49    TAYLOR-HUNYADI       ITO               MISHIMA
 15  321783    03JUN35    GONZALES             GUILLERMO         RICARDO
 16  328140    02JUN51    MEDINA-SIDONIA       MARGARET          ROSE
 17  346917    15MAR50    SHIEKELESLAM         SHALA             Y.
 18  356134    25OCT60    DUNNETT              CHRISTINE         MARIE
 19  423286    31OCT64    MIFUNE               YUKIO             TOSHIRO
 20  456910    24SEP53    ARDIS                RICHARD           BINGHAM
 21  456921    12MAY62    KRAUSE               KARL-HEINZ        G.
 22  457232    15OCT63    LOVELL               WILLIAM           SINCLAIR
 23  459287    05JAN34    RODRIGUES            JUAN              M.
 24  677890    24APR65    NISHIMATSU-LYNCH     CAROL             ANNE
```

For more information on the APPEND procedure, see the *SAS Procedures Guide*.

You can append data described by a view descriptor to a Version 8 data file just as you do with a Version 6 data file. However, when the FORCE option is used to append columns whose names do not match, any column names longer than 8 characters will truncated at 8 characters.

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS/ACCESS Interface to CA-DATACOM/DB Software: Reference, Version 8*, Cary, NC: SAS Institute Inc., 1999. pp. 170.

SAS/ACCESS Interface to CA-DATACOM/DB Software: Reference, Version 8

ISBN 1–58025–545–0

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

1st printing, October 1999

The Institute is a private company devoted to the support and further development of its software and related services.