In order to extend the analytical, reporting, and graphics capabilities of SAS® software to users whose data are stored in Data Base Management Systems (DBMS), SAS Institute has developed interfaces to some of the major DBMSs. The products currently available are SAS/IMS-DL/I® software for access to DL/I data bases, the SYSTEM 2000® DBMS interface procedures to access System 2000 data bases, SAS/DB2™ software for access to DB2 tables and SAS/SOL/DS™ software for access to SOL/DS tables. Currently under development is an interface to IDMS/R data bases.

SAS/IMS-DL/I software runs under the MVS/370, MVS/XA, VS1, and VSE operating systems. You can read and update DL/I data bases using either IMS/VS or CICS/VS. Under MVS, the access can be batch DL/I, IMS/VS Batch Message Processing (BMP), or CICS/VS Shared DL/I. Under VSE you can access the DL/I data bases in batch or through CICS/VS Multiple Partition Support (MPS). The type of execution is controlled by the value of the DLIRGNT option set on the OPTIONS statement. The DLIRGNT option defaults to DL/I if it is not specified.

There are two methods of accessing DL/I data using SAS software. They are the DATA step interface and the DLITEST procedure. The DATA step interface allows users to read and update DL/I data bases using the INFILE and FILE statements. The INFILE statement has options that may be used to define the DL/I-specific information required to build a DL/I call. Variables such as DBNAME, PSBname, CALL, SSA, and STATUS are used to construct a call that is executed by the INPUT or PUT statement. You can read and update data in a DL/I data base using the DATA step interface.

This DATA step reads an IMS data base, ACCOUNT, to produce a SAS data set containing checking account balances.

```
DATA ACCTREPT;
LENGTH SSA1 $11;
LENGTH SSA2 $9;
INFILE ACCTREAD DLI
  DBNAME=DBMNAME
  CALL=CALL
  SSA=SSA1.SSA2
  STATUS=STATUS;
DB='ACCOUNT';
FUNC='GN';
SSA1='CUSTOMER';
SSA2='CHCKACCT';
INPUT @12 CUSTNAME $CHAR40. @112 CUSTCITY $CHAR28. @140 CUSTSTAT $CHAR2. @238 BALANCE PD5.2;
IF _ERROR_ THEN DO;
  _ERROR_ = 0;
IF STAT = 'GE' THEN STOP;
ELSE ABORT ABEND;
END;
```

The following example updates the same IMS data base to change phone numbers of current customers.

```
DATA _NULL_;
SET NEW.PHONE;
LENGTH SSA1 $40;
INFILE ACCTUPDT DLI
  DBNAME=DBMNAME
  CALL=CALL
  SSA=SSA1
  STATUS=STATUS;
DB='ACCOUNT';
FUNC='GHU';
SSA1='CUSTOMER(SSNUMBER~CUSTNO)';
INPUT;
IF _ERROR_ THEN ABORT ABEND;
FILE ACCTUPDT DLI;
FUNC='REPL';
SSA1='';
PUT _INFILE- @;
PUT @172 NEWHPHON;
IF _ERROR_ THEN ABORT ABEND;
```

For users who want to manipulate their data interactively, there is PROC DLITEST. PROC DLITEST is a full-screen, interactive procedure that executes DL/I calls and displays the results of the calls plus any data returned by the call on the user’s screen. You can use it to retrieve specific segments from the data base or to view the segments sequentially. Segment Search Arguments (SSAs) can be used to define a segment or path of segments to be retrieved. If the PSB you are using has update authority, you may also use PROC DLITEST to update the DL/I data. Screen 1 shows an example of the data displayed by PROC DLITEST.

### Screen 1 PROC DLITEST Screen With Data From The ACCOUNT Data Base

For those of you who are concerned about the security aspects of using SAS/IMS-DL/I software, there are several security features that help protect your data. First, we fully support DL/I security. We do nothing in our interface code to compromise the security provided by IMS or CICS; we are merely constructing...
DL/I calls as any application program would do and using documented interfaces to send and receive requests to DL/I. In addition, we have implemented SAS options which you can set at installation time if you want to prevent all users from updating data bases with SAS/IMS-DL/I software. The options can be individually set for DL/I regions, DBB regions, S1I regions, and MPS regions. The options can be so set that they cannot be overridden. Thus you can restrict SAS/IMS-DL/I programs to read-only access if you want. We also have a user exit that you can use to examine all DL/I calls before they are executed. You can do special security validation, logging, or other monitoring and analysis of access to data bases for SAS/IMS-DL/I users.

The reports from our users who have looked at the performance of SAS/IMS-DL/I software compared to a high-level language DL/I program show that the SAS/IMS-DL/I programs are from 10 to 40 percent slower than high-level language programs. Because of the similarities in performance of a SAS/IMS-DL/I program and a high-level language BMP program, the fact that we are building standard DL/I calls that are processed through documented interfaces, a SAS/IMS-DL/I BMP job will have about the same performance impact on an online IMS system as a BMP written in a high-level language.

If you are interested in additional information on SAS/IMS-DL/I software, see Cummings (1987). This paper describes methods that may be used to make SAS/IMS-DL/I software less complicated for a novice DL/I user.

The next interface we will look at is our interface to SYSTEM 2000 DBMS. We currently have three procedures that run on either MVS or CMS and allow you to access SYSTEM 2000 data bases in either single or multi-user modes. The three procedures are PROC S2K, which extracts data from a SYSTEM 2000 data base into a SAS data set; PROC S2KLOAD, which loads data from a SAS data set into a SYSTEM 2000 data base; and PROC QUEST, which allows you to query and update SYSTEM 2000 data from the SAS Display Manager System.

If you need to analyze or graph your SYSTEM 2000 data, you can use PROC S2K to extract the data into a SAS data set where the full range of SAS procedures can be used to manipulate the data. PROC S2K allows you to select only the items you want to extract and to subset the data by filing in selection conditions in the WHERE statement. You can use the full-screen panels to define the items to be selected, and the SAS names to be used then save that information into a SAS data set, which can be used as input for a SAS job. This is particularly useful when you are extracting a large amount of data or you extract the same information frequently. Screen 2 shows the S2K procedure Extract Data Selection panel with items selected for extraction.

Screen 2 S2K Procedure Extract Data Selection Panel

If you have a SAS data set that you want to load or insert into a SYSTEM 2000 data base, you can use PROC S2KLOAD. The SYSTEM 2000 data base data sets must have been created, and the data base must have been defined using the SYSTEM 2000 DEFINE Language. The data base can be empty or can already contain data. You map the variables in the SAS data set to items in the SYSTEM 2000 data base, and the procedure inserts data from the SAS data set into the data base based on the mapping. You may specify a HOLD variable, which is used to build trees of items with the same HOLD variable value. PROC S2KLOAD also allows you to build the map data set interactively and use it as input for a batch job to do the actual load of data. Screen 3 shows the S2KLOAD Procedure Mapping Specification panel as you would complete it to load the Employee data base.

Screen 3 S2KLOAD Procedure Mapping Specification Panel

PROC QUEST executes SYSTEM 2000 query language statements from a SAS session. The procedure allows you to use display manager editing commands or the macro language to build and submit SYSTEM 2000 queries and view the results in the SAS output window or the SAS print file. You can use PROC QUEST to update data, produce reports, define data bases, or inquire on data in a data base. Screen 4 shows the results of a query using PROC QUEST.
The procedure uses SYSTEM 2000 passwords to govern the data that the user can see and modify. Additionally, we have separated the functions into different procedures, so the installation can better control their use.

The S2K and S2KLOAD procedures use the PLEX programming interface and will have performance data similar to that of a high-level language program. Features such as the WHERE clause and the item selection allow you to pass only a subset of the data. For more detail about the procedures, see Wills and Cummings (1987).

We have a new product, SAS/DB2 software, that interfaces to IBM's relational data base system called DB2 and that runs under the MVS/370 and MVS/XA operating system. Our interface has three new procedures, PROC DB2EXT, PROC DB2LOAD, and PROC DB2UTIL, plus an interface to the SAS/FSP® procedure FSEDIT.

PROC DB2EXT executes interactively and in batch to extract data from DB2 tables or views into either permanent or work SAS data sets. Using the WHERE clause, you can join or relate tables and views, or you can select only a specific portion of the data in the tables and views. You can supply your own SAS variable names, or you can use the system-created ones. Like the SYSTEM 2000 interface procedures, you can select the columns to be selected interactively and use this information as input for a batch job that actually extracts the data. Screen 5 shows the PROC DB2EXT Data Extraction panel used to build a temporary SAS data set WORK.EMPLRPIT.

PROC DB2LOAD executes interactively and in batch to load the extracted data into a DB2 table. You can specify the name of the SAS data set containing the data and the name of the DB2 table you want to create, and the procedure will create the table and load the data into it. If you want to create a DB2 table from data in a DL/I data base, a VSAM file, or other data set that can be read by a SAS DATA step, you can create a work SAS data set containing the data and then use PROC DB2LOAD to create the DB2 table. Screen 6 shows the PROC DB2LOAD Table Create/Load panel which will load table CORP.VACATION.

Once the DB2 table has been created, you can use PROC DB2UTIL to perform updates to it. You can insert, update, or delete tables using SAS data set input. You indicate which SAS variables map to DB2 columns and any WHERE clause specifications to be used in updating or deleting data. Screen 7 shows the PROC DB2UTIL Table Mapping panel used to update CORP.VACATION.
Screen 7 PROC DB2UTIL Table Mapping Panel

If you want to update rows of a DB2 table directly and you have SAS/FSP software, you can use PROC FSEDIT to edit the tables. You use the \texttt{TABLE=} option to define the table you want to edit and then use PROC FSEDIT commands to retrieve, search, and update the data using either the default screens or your own customized screens.

All of these procedures support DB2 security mechanisms and only access tables for which the users have been granted authority. We also have individual access modules for each procedure so that you can grant execute authority by procedure for additional control.

For those of you with questions about performance, we do use dynamic SQL calls. This was necessary to provide the flexibility that you would require. Options like COMMIT and LIMIT allow you to maximize the performance of the procedures, and you can also defer the CPU-intensive portions of the procedures to a batch execution.

We have a new interface product for CMS users called SAS/SQl-DS software. It provides an interface to IBM's SQL/DS and is very similar to our interface to DB2. SAS/SQl-DS software consists of three new procedures: PROC SQLEXT, PROC SQLLOAD, and PROC SQUdITIL, plus an interface to PROC FSEDIT. The procedures have analogous functions to their DB2 counterparts. The security is very similar to that of SAS/DB2 software also; we do nothing to compromise the SQL/DS security. If you would like to have additional information about the SAS/DB2 and SAS/SQl-DS products, see Wallace (1987).

We are currently involved in a project to develop an interface to IDMS/R developed by Cullinet Software, Inc. We are working on a procedure to extract data from an IDMS/R database into a SAS data set using views defined to the Logical Record Facility of IDMS/R. We hope to provide capabilities of updating IDMS/R data bases from SAS procedures in the future. If you would like more detailed information on the interface to IDMS/R, see Wills (1987).

Now that we have covered all of the SAS System interfaces to database management systems, I am sure that you are wondering how you could get us to develop an interface to your favorite DBMS. Probably the largest single influence on our decision is the results of the SASware Ballot. There is a section for data base interfaces, and we use that as an indication of the interest in various data base interfaces. If your favorite DBMS is not on the SASware Ballot, write a letter to our Technical Support Department and ask that it be added to the ballot. You can also write a letter to our Marketing Department to let us know what interfaces you need, or you can tell your DBMS vendor that you would like to have an interface to SAS software. If none of these methods are successful or are too slow, you can write your own interface using SAS Technical Report P-156, "INFILE Statement User Exits," which provides information on writing a DATA step interface.

Our future plans for DBMS interface development can be stated very simply - more interfaces, more powerful interfaces, and interfaces that are easier to use. We want to provide you with interfaces that will access the data directly without having to perform the intermediate extraction step and interfaces that inexperienced DBMS users can use. We also want to make our interfaces similar to each other so that the same processes are used to analyze IMS data, DB2 data, or any other DBMS's data.

\textbf{References}
