Introduction to SAS/DB2™ and SAS/SQL-DS™ Software

Emily P. Wallace, SAS Institute Inc., Cary, NC

For those of you who are not familiar with Database 2 or DB2 and SQL/Data System or SOL/DS, they are IBM’s relational data base management systems. DB2 runs under the MVS/370 or MVS/360 operating systems, and SOL/DS runs under the VM/SP operating system. Both DB2 and SOL/DS store data in tables made up of rows and columns and use the Structured Query Language, or SQL, to access the data.

SAS/DB2™ and SAS/SOL/DS™ software products were developed to provide SAS® users with a way to access and analyze their DB2 and SOL/DS data using SAS software. For the past several years, interfaces to these data base management systems were highly requested items on our SASwar ballot and we have also received numerous letters, phone calls, and verbal requests letting us know that you were interested in these interfaces. SAS/DB2 and SAS/SOL/DS software are very similar in function and will be discussed together because of this similarity. Differences between the products will be pointed out as the specifics of the software products are presented.

The DB2 and SOL/DS tables are similar to SAS data sets in concept; the rows and columns map nicely to SAS variables and observations. For that reason, interfacing the two software products became an easier job and gave us the ability to provide additional functionality.

The interface allows users who are executing their SAS sessions on the same CPU where DB2 or SOL/DS is executing to access tables and views using SAS software. For DB2 users, when the user starts one of the DB2 interface procedures, we attach a DB2 subtask to the user’s SAS session. We use the Call Attach Facility provided by IBM (PTF UF90214-tape 8506 for Release 1.0) to attach DB2 as a subtask. This means that we do not have to execute SAS software under the DB2 DSN command processor. We attach DB2 only for the duration of the procedure and disconnect it at procedure termination, so there is no DB2 overhead for the remainder of the SAS session.

For SOL/DS users, the process is easier. The user must have access to the SOL/DS machine and have previously run the SOUNIT EXEC to establish the name of the data base where the data reside. When the user starts one of the SOL/DS procedures, we simply construct the query and pass it to SOL/DS, who processes it and sends us a response. We format and display the response for the user.

Our interface uses dynamic SQL calls. We construct the call from user input and pass the call to DB2 or SOL/DS. They execute the request and return any requested data along with a return code indicating the success or failure of the request. For those of you who are wondering why we do not use static SQL calls for performance gains, realize that we know nothing about the table or view until you supply its name so our calls must be built as we go.

The DB2 interface consists of the following procedures:

- **PROC DB2EXT** - extracts data from DB2 tables into SAS data sets
- **PROC DB2LOAD** - creates and loads DB2 tables from SAS data set input
- **PROC DB2UTIL** - updates DB2 tables using SAS data set input
- **PROC FSEDIT interface** - allows PROC FSEDIT to update DB2 tables.

The SOL/DS interface consists of the following procedures:

- **PROC S QLEXT** - extracts data from SOL/DS tables into SAS data sets
- **PROC SOLLOAD** - creates and loads SOL/DS tables from SAS data set input
- **PROC SOLUTIL** - updates SOL/DS tables using SAS data set input
- **PROC FSEDIT interface** - allows PROC FSEDIT to update SOL/DS tables.

PROC DB2EXT and PROC S QLEXT execute in full-screen mode, line mode, and in batch. To invoke the procedure in full-screen mode, type

```plaintext
PROC DB2EXT;
RUN;
```

or

```plaintext
PROC S QLEXT;
RUN;
```

You then see the Data Access panel (Screen 1), which allows you to supply information about the table or tables from which you want to extract data. Under MVS, you fill in the name of the DB2 subsystem owning the table you want to access. The system uses the the default subsystem id of DB2 if you do not supply a subsystem name. Under CMS, there is no subsystem id because the SOUNIT EXEC has previously defined the data base machine you will access.

You then supply the name of the SAS data set where you want to store the extracted data in the OUTPUT SAS DATA SET field. The SAS data set can be either a permanent or work data set, and if no name is supplied, the default, WORK.DATAn, is used.

The OUTPUT SQL LIBRARY field allows you to specify the name of a member in the OS data set or the CMS file where the SQL SELECT statement used to extract the data will be stored. This field is optional and, if omitted, no query will be saved.

The SAVE SELECTIONS field allows you to specify whether or not you want the procedure to extract the data or just build the necessary information to do the extraction and place this information in the data set defined by the OUTPUT SAS DATA SET field. If it is set to Y, no data will be extracted. The SAVE SELECTIONS option is particularly useful when you have a large table. You can create the information to perform the extract interactively, save it in a SAS data set, and execute the actual data extraction as a batch job.

The UNIQUE SAS NAMES field indicates whether or not you want the procedure to generate unique SAS names from the column names. The column names are 18 characters, and the procedures can generate unique eight-character SAS names automatically or allow you to supply your own SAS names.

You must supply at least one authorization id and table name in the DB2 AUTORIZATION ID and TABLE NAME fields to get a full-screen list of the columns in the table and their descriptions. The names you supply define the tables, views, or a combination of both tables and views that contain the data you want to extract.
The authorization id is an installation-defined name that specifies the kind of actions that may be performed on a table using that authorization name. The TABLE NAME field tells the name of the table or view that contains the data you want to extract. Any combination of five tables and views may be used. We have elected to limit the number of tables to five based on IBM's recommendation that the 'practical limit' of the number of tables joined by one select statement is four.

**Screen 1** Data Access Panel for the DB2EXT or SQLEXT Procedures

Once you have completed the Data Access panel, press ENTER. This causes PROC DB2EXT or PROC SQLEXT to use the information you have supplied to validate that the table exists and that the authorization id has read authority for the table. Once the validation is done, we display the Data Extraction panel (Screen 2), which lists the column names and data types for each table you have supplied on the prior screen. Notice that the format is the SAS format name for DB2 or SOLIDS data type.

For each column you want to extract, type an S in the FUNC field for that column name or fill in the SAS NAME field with a valid SAS name. If you use the S function to select a column for extraction and do not supply a SAS name, the procedure fills in the SAS name with the first eight characters of the column name. You can change the default SAS name supplied by the system if you want.

If you have selected a column and decide you do not want to extract it, put a blank in the FUNC field or put blanks in the SAS NAME field for the column. You may change the format of any column displayed on the screen. This causes the data in the columns you have changed to be reformatted to the formats you have supplied when the data are extracted.

If you know that a column in the table contains dates, you can specify any of the SAS date formats and the data will be converted to SAS dates when extracted. If the date is not stored as a YYMMDD value in the table, follow the SAS date format with a slash and the format that describes the data as it is stored.

**Screen 2** Data Extraction Panel for the DB2EXT or SQLEXT Procedures

When you press the ENTER key, SAS names are generated for columns you have selected but not supplied a SAS name. You can keep making additions and corrections to the panel until you have the columns and their names and formats as you want them.

The WHERE CLAUSE portion of the screen can be used to specify a condition or set of conditions to use in selecting the data. Only those rows that meet the conditions described in the WHERE clause will be selected. The WHERE clause can also be used to join or relate two tables. You must specify the names of the two columns whose data values must be equal. This identifies the pair of rows (one from each table) that will be joined, and the joined row will be used for data extraction.

If you want to view the SQL SELECT statement that was built from the information supplied on the Data Access panel and the Data Extraction panel, type EDIT on the command line. The SELECT statement that is displayed on the SQL Entry/Edit screen (Screen 3) will be used to extract the data from the table or tables. If you want to make any changes to the SELECT statement before the data are extracted, you may do so. If you supplied a file name for the OUTPUT SQL LIBRARY field on the Data Access panel, this SQL statement will be saved in the desired file.
After selecting the columns to be extracted using either the Data Extraction panel or the SQL Entry/Edit screen, press the END key. This initiates the extract process unless you have chosen the SAVE SELECTION option. You will be returned to the Data Access Panel where you will see statistics indicating the number of variables and observations in the output SAS data set. You can then continue your session to extract additional data or terminate the procedure and analyze the data you have just extracted.

If you want to bypass the Data Extraction panel and enter your own SQL statement or include it from an external file, you should leave the AUTHORIZATION ID and TABLE NAMES fields blank on the Data Access panel. A blank SQL Entry/Edit screen will be displayed, and you can type the SQL SELECT statement you want to execute or use the INCLUDE command to copy a SELECT statement from a previously allocated PDS. The statement may be edited as desired, including adding or deleting columns and WHERE clauses.

For users who want to bypass the panels entirely, PROC DB2EXT and PROC SOLEXT support interactive-line-mode execution. If the terminal you are using supports full-screen procedures, you can use the RUN command to indicate whether or not full-screen or line mode access is used. If you enter the RUN command immediately after the PROC statement, you will enter full-screen mode and can use the panels to extract the data. If you enter other statements, such as TABLE, LIST, or SELECT, you will be using interactive-line-mode access. The line mode execution will allow you to enter or include any valid SQL SELECT statement and have the data extracted into a SAS data set defined by the OUT option supplied in the PROC statement.

The LIST statement allows you to view information about the columns you have selected or the columns contained in a table. The LIST ALL command displays the column names, the SAS variable names, and the SAS formats for the columns you have selected for extraction. The LIST tablename command displays the name of all columns in the table or view that the authorization id is permitted to read. You can use the RENAME statement to change the SAS names and the FMT (format) statement to change the output formats. If you want to change the columns that have been selected, you may type in a new SELECT statement. When you issue the RUN statement, the extraction takes place and the SAS data set is built (Screen 4).

```
2>
proc db2ext run=testing;
   notes: temporary table has not been created.
   notes: temporary table has been created.
   proc demo db2util file db2demoemplinfo where vacation>0;
   list all;
   select employee, lastname, firstname, hiredate, vacation, department from
   db2demoemplinfo where vacation > 0;
   list all;
```

Screen 4 PROC DB2EXT Line-Mode Execution

The statements used in batch execution are identical to those used in interactive line mode. One additional feature that is important in batch mode is the SAVESEL option, which allows you to use full-screen or interactive line mode to define the columns to be extracted, establish appropriate SAS names, and supply any WHERE clause subselection criteria. This selection information is stored in a SAS data set and can be used as input for a batch mode execution. This allows you to take advantage of the interactive portions of the procedure to define the data to be extracted and defer the CPU-intensive data extraction to a batch job. The batch mode also accepts all SQL SELECT statements as input.

The DB2EXT and SOLEXT procedures provide users with a facility to bring data from the DB2 or SOLIDS environments into the SAS environment, but it does not help users who have their data in a SAS data set and want to send them to DB2 or SOLIDS. To meet this need, we have developed PROC DB2LOAD or PROC SQLLOAD to create and load a table and PROC DB2UTIL or PROC SQLUTIL to modify an existing table.

To create and load a table using the full-screen panels, type

```
proc db2load;
run;
```

or

```
proc sqlload;
run;
```

to view the Data Access panel (Screen 5). The INPUT SAS DATA SET field contains the name of the permanent or work SAS data set that contains the data you want to use as input for the table creation. You specify the name of the table you want to create in the TABLE NAME field.

The COMMIT field specifies the number of SQL INSERT calls issued between COMMIT calls and can be used as a performance tuning tool. The LIMIT field is used to specify the maximum number of INSERT calls to be executed by the procedure. If you want to create a table with a large number of rows, you might execute a test job with the limit set to ten to verify that the table will be loaded correctly before actually running the job to create and load the entire table.

The MAP DATA SET field defines either an input or an output SAS data set containing the mappings of SAS data set variables to DB2 columns that will be used to load the table. This data set can be built interactively and used as input for a batch execution to actually load the data. If you specify a file name for the ERROR LOG FILE field, any SQL statements that are not inserted due to SQL errors will be written out to this file. The errors can be corrected and the file can then be incuded in another execution of the procedure.
SCREEN 5  Data Access Panel for the DB2LOAD or SOQLLOAD Procedures

Once the information is correctly entered on the Data Access panel, press ENTER to view the Table Create/Load panel (Screen 6). This displays the list of variables in the SAS data set and the information that will be used to create the DB2 or SOQL/DS table. The column names and formats may be changed to any values that are valid. The NULLS? field can be used to denote fields that may not contain null values in the table. The default for the NULLS? field is Y, which allows null values in the table. The IN field, which defines the tablespace or database in which the table will be placed, is particularly important to SOLIDS users. In DB2, if you do not specify a tablespace or database name, the default value DATABASE DSNDB03 is used. However, in SOL/DS the default is a private DBSPACE of the authorization id or userid. Unless the authorization id owns a DBSPACE, the procedure will receive a return code from SOLIDS indicating an error condition.

SCREEN 6  Table Create/Load Panel

When all of the information is correctly specified, type LOAD on the command line and press the ENTER or END key. This causes the table to be created and the data to be loaded. After the load process has completed, the Data Access panel is redisplayed showing statistics that tell how many records were inserted and how many errors were encountered.

PROC DB2LOAD and PROC SOQLLOAD have a SQL Entry/Edit screen similar to that of PROC DB2EXT and PROC SQL1EXT that allows you to type in any SQL command except a SELECT statement and execute it directly. The COMMIT and LIMIT values do not apply to commands typed in using the SQL Entry/Edit screen.

Line mode and batch mode of PROC DB2LOAD and PROC SOQLLOAD allow you to perform the same functions as the full-screen execution.

For those users whose tables have already been created, PROC DB2UTIL and PROC SOLUTIL allow them to modify their tables using a SAS data set as input. To invoke the procedure, type

PROC DB2UTIL;
RUN;

or

PROC SOLUTIL;
RUN;

You will see a Data Access panel (Screen 7) that is very similar to the one displayed in the DB2LOAD or SOQLLOAD procedure. The major difference is the FUNCTION field, which specifies the kind of modification you want to perform. You may insert, update, or delete rows of the table, but you may perform only one function at a time using the full-screen panels. The CORRELATION NAME is used to specify a name that may be used as an alternative name in a WHERE clause for updates or deletes.

SCREEN 7  Data Access Panel for DB2UTIL and SOLUTIL Procedures

If you have chosen the insert function, you will see the Table Mapping Panel after pressing ENTER. You indicate the mapping of SAS variable names to column names for the data insertion. If you want to view to column names in the table, type DISPLAY in the SELECT field, position the cursor where you want the display to begin and press ENTER. You can use either the column name or its number to indicate the mapping. When you have mapped all variables to columns, type UPDATE on the command line and press the ENTER or END key. The inserts will be done, and you will be returned to the Data Access panel, which will display statistics to show how many rows were inserted.
If you choose the update function, you will also see the Table Mapping panel (Screen 8). You will map the variables in the SAS data set to the columns they will update. The WHERE Entry panel is used to specify the variables which will determine which row to update. The UPDATE command is used to perform the updates.

```
Screen 8 Table Mapping Panel of PROC DB2UTIL
```

If you are deleting rows from the table, you will see the WHERE Entry panel, which allows you to specify the variables that will determine which rows will be deleted. The UPDATE command causes the rows to be deleted.

All DB2UTIL and SOLUTIL functions are available in line mode and batch execution.

I have shown you tools for transferring your data back and forth between SAS software and DB2 or SOL/DS, but there are users who would like to update their tables directly from their SAS session. For those users, we have developed an interface between DB2 or SOL/DS and PROC FSEDIT, a SAS procedure that includes full-screen editing capabilities. You invoke PROC FSEDIT in the normal manner except you specify a TABLE= option instead of a DATA= option. The TABLE= option identifies the table or view that you want to edit. From that point on, you have the full scrolling, updating, and searching capabilities provided by PROC FSEDIT. You can insert rows into the table, update existing rows, or delete unneeded rows. You can use the FIND and SEARCH commands to locate records with a certain value for a given variable. You can create your own customized screens to display the data. The only portion of PROC FSEDIT that is not permitted for DB2 and SOL/DS tables is the NEW= option.

I want to briefly address some security issues about which you might be concerned. We fully support DB2 and SOL/DS security mechanisms; we do nothing to circumvent or compromise their security. The SAS/DB2 and SAS/SOL/DS software products were developed with separate procedures for each function to allow sites to control which users can perform which functions. Each procedure has a separate plan or access module so that access to each procedure can be granted to only those users who need to perform the function. In the DB2UTIL and SOLUTIL procedures there are actually three plans or access modules so that each function can be individually controlled.

```
DB2 INTERFACE  DB2 TABLE - CORP. VACATION
SAS DATA SET - WORK, VACUSED

SELECT: MAP
MAF TO COLO\N
SAS WHERE

COMMAND ->

MAPE COLUMN SAS BASE FORMAT
EMPLOYEE DATE 6.

COMMAND ->

Select: DISPLAY

NEW COLUMN TYPE
01 EMPLOYEE DECIMAL(4,0)
02 LASTNAME CHAR(10)
03 FIRSTNAME CHAR(20)
04 ADDRESS DECIMAL(6,0)
04 SOCIAL CHAR(11)

COMMAND ->

Select: WHERE

SAS WHERE STATEMENT: employee + amount
```

To use the software, the site must license base SAS software and the SAS/DB2 or SAS/SOL/DS software products. If a site wants to use the PROC FSEDIT interface, SAS/FSP® software must also be licensed. SAS/DB2 software will work with both Release 1 and Release 2 of Database 2. DB2 Release 1 must have PTF UP90214 from tape 8506 (the Call Attach Facility) installed. SAS/ SOL-DS software requires Release 3.5 of SOL/DS.

SAS, SAS/FSP, and SASware Ballot are registered trademarks of SAS Institute Inc., Cary, NC, USA. SAS/DB2 and SAS/SOL-DS are trademarks of SAS Institute Inc.